Emergence of Practical Language for Locations in a Simulated Warehouse through language games

Otto Hantula

Helsinki May 27, 2019

MSc Thesis

UNIVERSITY OF HELSINKI
Department of Computer Science

HELSINGIN YLIOPISTO — HELSINGFORS UNIVERSITET — UNIVERSITY OF HELSINKI

Tiedekunta — Fakultet — Faculty	Koulutusohjelma — Studieprogram — Study Programme
Faculty of Science	Computer Science

Tekijä — Författare — Author
Otto Hantula

Työn nimi — Arbetets titel — Title

Emergence of Practical Language for Locations in a Simulated Warehouse through language games

Ohjaajat — Handledare — Supervisors
Hannu Toivonen and Simo Linkola

Työn laji — Arbetets art — Level	Aika — Datum — Month and year	Sivumäärä — Sidoantal — Number of pages
MSc Thesis	May 27, 2019	63

Tiivistelmä — Referat — Abstract

Emergence of language grounded in perception has been studied in computational agent societies with language games. In this thesis language games are used to investigate methods for grounding language in practicality. This means that the emergence of the language is based on the needs of the agents. The needs of an agent arise from its goals and environment, which together dictate what the agents should communicate to each other.

The methods for practicality grounding are implemented in a simulation, where agents fetch items from shelves in a 2D grid warehouse. The agents learn a simple language consisting of words for spatial categories of xy-coordinates and different types of places in the warehouse environment.

The language is learned and used through two novel language games called the Place Game and the Query Game. In these games the agents use the spatial categories and place types to refer to different locations in the warehouse, exchanging important information that can be used to make better decisions.

The empirical simulation results show that the agents can utilise their language to be more efficient in fetching items. In other words the emerged language is practical.

ACM Computing Classification System (CCS):
Computing methodologies → Artificial intelligence → Distributed artificial intelligence → Multi-agent systems
Computing methodologies → Modeling and simulation → Simulation types and techniques → Agent / discrete models
Computing methodologies → Machine learning → Learning settings → Online learning settings

Avainsanat — Nyckelord — Keywords
multi-agent systems, language games, language emergence, computational agents

Säilytyspaikka — Förvaringsställe — Where deposited

Muita tietoja — övriga uppgifter — Additional information

Contents

1 Introduction

Computational agents situated in a shared environment can coordinate their actions through communication. However, what exactly the agents should communicate to each other is not always known in advance to the programmer. In this case it would be useful, if the agents could independently adapt their communication to fit their needs.

In this thesis the agents' communication method is a language, that emerges in the agent society through the agents' experiences and social interaction. The language consists only of individual words, that refer to certain meanings. There is no grammar and the agents don't construct sentences.

The meanings are such that they can be used to refer to certain objects in the agent's environment. Suppose for example that the agents are in an environment with two blue objects and one red object. If an agent wants to draw the attention of the other agents to the red object, it can do so by communicating the meaning "red".

Even though the language in this thesis is quite simple, its emergence is complex, because the agents learn individually. Each agent has its own meaning structure and lexicon, which have been learned through personal experience and are not directly accessible to other agents. The agents can infer what others mean by certain words only from the context the words are used in. Quine's example [1] of a linguist translating a new language illustrates the difficulty of such a task. The linguist is in the company of a native speaker, when a rabbit scurries by and the native says "Gavagai". The linguist cannot directly infer what "Gavagai" means, because there are several possible interpretations including "animal", "rabbit" and the colour of this particular animal (e.g. "white").

One of the greatest advantages computational agents have over humans is that they can directly transmit information to each other without having to encode and decode it through language. In humans this would correspond to a person sending her thoughts to another person via some kind of telepathy. Even among computational agents there are some situations where this is not possible. First, the internal representation of information might be too large to transmit if the agents have a limited bandwidth for communication. Second, the agents might have been implemented differently, so that they can't understand the information representations of each other. Third, there might be a need for the computational agents to work alongside humans, obviously making direct transmissions impossible with current

technology. In these cases communication might become possible if the information is represented as meanings with words linked to them. When an agent wants to communicate a meaning, it can just transmit the word linked to it, for example, by saying the word out loud.

In this thesis a language is deemed practical if it helps its users in achieving their goals. Thus what kind of meanings the language should be able to convey is dependant on the agents' goals. For a group of robots exploring a maze, meanings for directions, different kinds of junctions, and dead end might be useful. If the robots are instead playing soccer, meanings for defending, attacking, and passing might come in handy.

In recent years practical language emergence has been studied utilising neural networks [2, 3, 4]. Neural networks have several drawbacks, including their slow convergence and black box nature. In contrast simpler, explicit language models learned through language games are used in this thesis.

Language games provide the agents with a framework for language learning and usage, including a script to follow in their verbal communications. An example of a language game is the Guessing Game [5]. The Guessing Game has two participants: the speaker and the hearer. The speaker chooses a topic for the game, which might for instance be an object or another agent, and verbally describes it. The hearer, based on the verbal description, tries to infer the topic and indicates its guess to the speaker for example by pointing. If the hearer guessed correctly, the game succeeds. Otherwise the game fails. The agents can update their lexicons based on the outcome of the game.

Previous research on language games has focused on how a language might emerge and how different mechanisms affect the emergence. The agents' goal in these studies has been to learn a language in order to succeed in the language games they play. The agents have not had goals outside of the language games.

Diverging from previous research, the aim of this thesis is to lay some groundwork for research on language games grounded in practicality. The problem is approached with the following research question:

How might language games be grounded in practicality?

This means that it isn't enough for the agents in this thesis to be good only in

language games. They also have to be able to utilise the language as they act in an environment. This raises questions like when and how the agents should communicate. Although these could be learned, in this thesis they are predetermined. Instead the agents learn what is communicated, i.e. which meanings they try to communicate to each other in the language games.

The research question is approached both on a conceptual level and through empirical computer simulations. On a conceptual level the problem of grounding language in practicality is described generally and different properties of agents that make practicality grounding possible are introduced. To test and analyse an example implementation of these properties, an experiment is devised where agents fetch items in a simulated warehouse environment.

The rest of the paper is organised as follows. Section 2 discusses grounding language in perception, which serves as the problem setting for this thesis. The contributions of this thesis begin in Section 3, where the problem of grounding language in perception is extended to include grounding in practicality. Some abilities for the agents are also suggested to make grounding in practicality possible. Section 4 describes the experiment setting for the computer simulation experiments. The implementation of the simulations is detailed before the agents have any language abilities. Section 5 reports results from experiments where the agents act without language, establishing a baseline. In Section 6 the agents are endowed with the ability to use language. For this two new language games are introduced, which are used to conduct experiments to see if a practical language emerges in the agent society. In Section 7 different properties of the language are investigated further through empirical experimentation. Section 8 discusses related work. Finally Section 9 ends the thesis with conclusions and future work.

A paper summarising the work presented in this thesis has been published in the proceedings of AISB 2019 [6].

2 Grounding language in perception

In this thesis each agent has its own internal meanings and a lexicon that associates words with them. The meanings arise from observations of the environment and the lexicons emerge as a consequence of verbal interaction between agents. Both of these cases require some perceptual abilities from an agent, e.g. sight and hearing. Therefore it can be said that a language that emerges through these means is

grounded in perception.

The problem of grounding language in practicality is deeply embedded in perceptual grounding, which is why perceptual grounding needs to be discussed first. As a matter of fact, it would be safe to say that the practicality grounding presented in this thesis is merely an extension to perceptual grounding. In this section grounding language in perception is discussed through existing work.

The rest of this section is geared towards giving the reader a general idea of how the problem setting for language emergence is defined and approached in this thesis. First, the agent society as a whole is discussed. Then, different abilities an individual agent needs for language learning are described, and different kinds of meanings an agent might possess are outlined. Next, the main mechanism for language emergence in this thesis, i.e. language games, is considered. Finally, a model for discrimination meanings and a lexicon are described, both of which are later used in the experiments of this thesis.

2.1 Agent society

The agent society where the language emerges is a multi-agent system. There are multiple definitions for a multi-agent system, for example, Shoham and Leyton-Brown describe a multi-agent system as follows: "Multi-agent systems are those systems that include multiple autonomous entities with either diverging information or diverging interests, or both." [7]

This thesis considers agent societies where a language emerges in a distributed fashion without central control. Every agent learns separately and chooses its actions itself. It is important to note, that although all agents choose their actions through the same predefined rules, the choices are based on agent-specific knowledge and state. Due to the autonomous nature of the agents, the emergence of language is a bottom-up process.

The agents can only observe their immediate surroundings, so they are only partially aware of the state of the whole environment. For this reason the emergence of language is based on local interaction between agents. For the agents to learn a mutual language, they must have at least partially overlapping observations of the of the world. For example, the agents could see the same object at the same time. Through these mutual experiences they can learn common words. Once the agents have learned some words, they can have conversations even without overlapping

observations. For example, if agents A and B both have experiences about socks and colours, A could inform B about the colour of its socks, even though B would have never seen these particular socks in its lifetime. To the author's knowledge previous research on language games has not exploited this property of language. Instead, the agents have always had at least a partially overlapping observation of the world.

For simplicity's sake, although the purpose of the communication in this thesis is the coordination of the agents, the agents don't explicitly aim to cooperate with each other. The agents are selfish and act only based on their personal goals. Considering other agents' goals would require an agent to model the "minds" of its peers [8]. An agent is also always truthful in its communication, even though being deceitful might aid it in its goals. Modelling trust is outside of the scope of this thesis.

2.2 Processing layers

The agents need to be able to observe their environment and interpret language in order for a language grounded in experiences to emerge in a multi-agent system. The agents need to be able to associate perceptual observations with words and vice versa. Therefore the process that associates observations and words has to be bidirectional. Steels [5] describes an architecture, in which this process involves three layers: a perceptual layer, a conceptual layer and a lexical layer. The focus of this thesis is on the conceptual and lexical layers.

The perceptual layer transforms the signals from an agent's perceptual sensors into a form the agent can process internally. For example object detection could be performed on an image captured by the agent's camera. For each object detected, different features such as size, colour, and position could be calculated. These features would at this point only be expressed as numerical values.

The conceptual layer deals with conceptualising the information received from the perceptual layer. The agent could have a concept for "red", defined by boundaries for colour values. Or it could have a concept for "left", meaning an x-coordinate that is smaller than the middle x-coordinate of the image. Concepts such as "red" and "left" are meanings that the agent can associate with words.

The speaker chooses what meaning it wants to express in the conceptual layer. For example if the speaker wants to draw the hearer's attention to a certain object, it has to choose a meaning that somehow discriminates the object from other objects.

Let's consider a situation with two blue objects, o_1 and o_2, and a red object, o_3. All objects are the same size. The speaker wants to draw the hearers attention to the object o_3 and can only express size or colour. While trying to decide what meaning to express, the speaker could notice that size does not discriminate o_3 from the other objects, so it is of no use. On the other hand o_3 is the only red object, so the speaker could choose "red" as the meaning to convey.

The hearer's task in the conceptual layer is the opposite. It's goal is to associate a meaning with an object or objects. In the situation described above the meaning for which a corresponding object needs to be found is "red". Luckily there is only one red object, so the association can be done unambiguously. If the speaker had used the meaning "blue", the hearer wouldn't have had any way to know whether the speaker is referring to object o_1 or o_2 (or both).

The lexical layer maintains associations between meanings and words. The associations can strengthen and weaken depending on in which context the agent hears the words being used. In the lexical layer the speaker finds a suitable word associated with the meaning it chose in the conceptual layer to be uttered. The hearer tries to interpret the word it hears by linking it to a meaning.

The perceptual layer is mostly relevant for agents that perceive the environment through sensors that output signals which require interpretation (e.g a camera or a microphone). In the simulations conducted in this thesis, the agents get the observations directly in a form that can be used by the conceptual layer.

Additionally, Steels describes a syntactic layer that enables constructing and interpreting structures containing multiple words such as sentences. The syntactic layer is out of the scope of this thesis.

2.3 Meanings

The agents could have many kinds of meanings, but only meanings that can be used to categorise objects are considered in this thesis. Formally, given the set of all objects O, a meaning m can be used to define a subset $O_m \subset O$. In Section 2.2 the set of all objects consisted of three objects $O = \{o_1, o_2, o_3\}$, from which only o_3 was red. The meaning "red" would define a subset $O_{red} = \{o_3\}$. The objects o_1 and o_2 were blue, so the meaning "blue" would define the subset $O_{blue} = \{o_1, o_2\}$.

Meanings can be absolute or relative. For example, let's take an object that is 140cm in length. An agent could have an absolute meaning, or a category, for objects

between 130 and 150cm. The 140cm object would always fall into this category regardless of the context. What about meanings "short" and "tall"? Clearly, if the object happened to be an adult man, he would be considered short. On the other hand if the object was a girl of age five, she would be considered tall. The meanings "short" and "tall" depend on the context and are therefore relative.

How should an agent deal with relative meanings then? In this thesis context-oriented scaling [5] is used, where an object's features are scaled (i.e. normalised) based on the context. Let's define the context as a subset of all objects $C \subset O$. Now if an object belongs to C, its feature, e.g. length, can be normalised with respect to all the objects in C. In this thesis features are normalised to the range $[0, 1]$ [1]. If C is all 5-year-old girls, normalising 140cm would result in a value clearly larger than 0.5. In the context of adult men, the normalised value would be clearly smaller than 0.5. By having length categories for below and above 0.5, an agent could have meanings somewhat corresponding to short and tall. Due to normalising, an agent can learn relative meanings, which can be used in many kinds of different contexts.

Meanings can of course also be formed in many other ways than just dividing a feature into value ranges. A meaning could be a combination of value ranges for different features [5]. An agent could also form prototypes, that would categorise objects by how close an object's features are to certain values (see Bleys et al. [9] for an example in the domain of colours).

There is a distinction to be made between meanings that are linked to a specific object and meanings that define a category of objects. For example, you might own a chair that you are so fond of, that you have decided to give it a name. Let's say you have named it "Bob-the-chair". "Bob-the-chair" now only means one object, your beloved chair. In other words from the set of all objects O, "Bob-the-chair" would define a subset containing exactly one specific object. In contrast the meaning "chair" defines a category of objects. "Chair" would define a subset from O containing all chairs. The focus in this thesis is on the latter kind of meanings, because they allow for more flexibility in communication. The benefit of general, category defining meanings can be seen for instance if the agents are moved to a new environment. If the agents would have just named specific objects in their old environment, they would have to create a new language from scratch in the new environment. With more general meanings, in the best case scenario, the agents

[1] Features values are assumed to be interval variables. Categorical or nominal features are not considered.

wouldn't have to make any adjustments to their language at all.

One can also imagine a dichotomy between functional and non-functional meanings. A functional meaning represents what the function of an object is. Consider a hammer for instance. Oxford Dictionary of English provides the following definition for hammer: "a tool with a heavy metal head mounted at right angles at the end of a handle, used for jobs such as breaking things and driving in nails" [10]. The latter part of the definition consists of a functional meaning, which describes what a hammer is used for. All the other meanings described so far in this chapter have been non-functional meanings. The length of an object only describes a physical feature of an object, not what its function is, and is therefore a non-functional meaning. Due to the difficulty of having an AI understand and learn functional meanings, this thesis considers only non-functional meanings. Non-functional meanings can already be extracted from physical objects with machine vision, as was done in the Talking Heads experiment [11].

2.4 Language games

Language games are an abstraction of language use created by Wittgenstein [12]. They were originally used to illustrate language use between humans. Wittgenstein describes the builder's language game as an example. In it a builder A and an assistant B have a language, that consists of the words: "block", "pillar", "slab", and "beam". A calls the words out and B brings the corresponding stones.

In this thesis language games are the main mechanism for language emergence. Agents learn meanings and align their lexicons by playing language games with each other. Diverging from previous research, in this thesis language games are also used to transfer important information, which the agents can use to make better decisions about how to act.

Steels [13] was the first one to use language games in a computational multi-agent system. Steels showed that a language consisting of words and meanings can emerge spontaneously in an agent society as a consequence of language games [14]. In Steels' experiments the agents started without meanings or words. The agents learned new meanings and words associated with them by repeatedly playing language games with each other. Words spread in the society only through local games between two agents.

A language game can involve multiple agents in different roles. Up to now research

in computational multi-agent systems has focused on games between just two agents, other one being the speaker and the other one the hearer [5, 15, 16, 17]. The speaker's goal is to express a certain meaning with an utterance. The hearer attempts to interpret the meaning behind this utterance. In Section 6.3 a language game called the Query Game is introduced, which involves one speaker and multiple hearers.

In language games the communication is not limited to verbal utterances. The agents can additionally have non-verbal communication methods. For example Spranger et al. [18] conducted experiments with humanoid robots, that can use their hands for pointing. This ability is used by the speaker to indicate to the hearer which object it was referring to with its utterance. [2]

Language games provide agents with a framework for their communication. Due to having a common framework, the agents don't have to consider all the practically infinite interpretations that an unknown word can have in any given situation. For example in the Guessing Game [5], the goal of the speaker is to give a verbal hint of an object, so that the hearer can identify which object is being talked about. Therefore the hearer can assume that the the speaker's utterance refers to a specific object and is something that can discriminate this object from other objects.

Included in the framework is a script for the agents to follow in their interaction. In the variation of the Guessing Game used in the Talking Heads experiment, agents observe shapes such as triangles, squares, and circles on a whiteboard [11]. An agent perceives the whiteboard through a camera, which can be used to look at different parts of the whiteboard. The context for the Guessing Game is determined by which part of the whiteboard the agents are looking at. The agents can perceive different features for the shapes including colour, average gray-scale and position. The speaker picks a shape as the conversation topic and chooses a word describing one of the features of the topic. The hearer tries to guess which shape the speaker is referring to. Below is the script for their interaction (assuming that the agents are already looking at the same part of the whiteboard):

[2] The robots could not infer which object was being pointed at by the other robot through visual information alone. Instead they sent the xy-coordinates of the objects to each other directly.

1. The speaker randomly picks the conversation topic from the set of shapes it perceives.
2. The speaker chooses a meaning that discriminates the topic from the other shapes in the perceived context and utters a corresponding word.
3. The hearer interprets the word, guesses what the topic is and points to it.
4. If the hearer is not able to come up with a guess or points to the wrong shape in step 3, the speaker points to the topic.

The Guessing Game described above is an example of a language game with feedback. The pointing gives both agents feedback on the success of the game. If the hearer points to the wrong shape, or fails to point at all, the speaker knows the game failed. If the speaker has to correct the hearer by pointing to the correct topic, the hearer knows the game failed. If the hearer points to the correct topic, both agents know the game succeeded. There are also language games where the agents don't get any feedback on the success or failure of the game. Smith [16] has shown that a language can emerge even without feedback.

In this thesis two new feedbackless language games are introduced: the Place Game and the Query Game. With these games the agents learn meanings to refer to different locations in their environment and communicate useful information to each other. The games are described in detail later in Section 6.

2.5 Language models

Language emergence requires the agents to have dynamic models for maintaining meanings and lexicons. In this section models, that are the basis for the implementation of our agents, are described.

2.5.1 Discrimination trees

Discrimination trees [19] can be used to categorise objects based on a specific feature. The nodes in a discrimination tree are categorisers, that define a value range. A node's range is always contained in its parent's range. Every node, or categoriser, is a separate meaning that an agent can associate with a word. A categoriser that defines the value range $[i, j]$ for feature f is notated with $[i\text{-}j]_f$.

Let's consider an agent which perceives an object's size, s, as a normalised value in range $[0, 1]$. The root of the discrimination tree would in this case be all values covering categoriser $[0\text{-}1]_s$. Dividing this range into two halves results in categorisers $[0\text{-}0.5]_s$ (small) and $[0.5\text{-}1]_s$ (large), which are the children of the root node. The categoriser $[0\text{-}0.5]_s$ could be further divided into categorisers $[0\text{-}0.25]_s$ (very small) and $[0.25\text{-}0.5]_s$ (smallish) and so on. This tree is depicted in Figure 1. In this thesis the value range of a categoriser is always divided into two halves, although this is not required by discrimination trees.

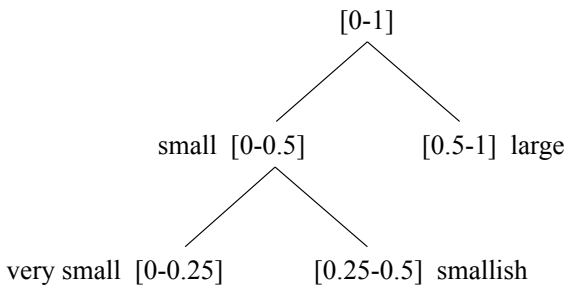


Figure 1: An example of a discrimination tree.

Agents can use discrimination trees to discriminate objects in the conceptual layer discussed in Section 2.2. For an example, let's consider a context with three object o_1, o_2, and o_3, with sizes 0.2, 0.6, and 0.8 respectively. An agent can discriminate the object o_1 from the other objects by using the categoriser $[0\text{-}0.5]_s$ (or $[0\text{-}0.25]_s$), because o_1 is the only object, whose size is in that categoriser's range.

In this context objects o_2 and o_3 can't be discriminated unambiguously with the discrimination tree described above, because it doesn't have a categoriser that would contain only one of them. If the agent had a need to discriminate between o_2 and o_3, it could grow the discrimination tree by dividing the categoriser $[0.5\text{-}1]_s$ into categorisers $[0.5\text{-}0.75]_s$ and $[0.75\text{-}1]_s$. With the new categorisers both o_2 and o_3 can be discriminated unambiguously from all other objects. Growing the discrimination tree is one way for the agents to gain new meanings.

2.5.2 Lexicon

In this thesis lexicon is defined as a model that maintains associations between meanings (e.g. categorisers in a discrimination tree) and words. It is used to both

transform a meaning into a word and a word into a meaning.

Steels [5] presents a lexicon, which stores a score between 0 and 1 for each meaning-word pair. The score represents how successful a pair has been. The higher the score, the more success has been had using the pair in language games,

A word can be associated with multiple meanings (ambiguity) and a meaning can be associated with multiple words (synonymy). See Table 1 for an example lexicon.

	webe	*fomodu*	*ranu*	*mamebo*
$[0\text{-}0.5]_s$	0.3	0.1	0.8	-
$[0.5\text{-}1]_s$	-	-	0.2	0.6

Table 1: An example lexicon.

Steels describes update mechanisms based on the success or failure of a game round. In this thesis the language games don't have feedback, so the agents don't know if the game succeeds or fails. Instead an update mechanism is used, where the hearer always updates the association between the word used by the speaker and the interpreted meaning. When the score of a meaning-word pair is updated, it is incremented by δ. For lateral inhibition, the score of all other pairs that contain either the meaning, or the word, is decremented by δ. In this thesis δ is set to 0.1. The score is clamped between 0 and 1.

Only the hearer updates the meaning-word association, because this was detected to work better while building the experiments. However, no formal experiments were conducted to investigate the issue further. The author speculates that it is better to only try to learn what words other agents are using and not to confuse the learning process also with words oneself uses. Image for example an agent society, where all agents but one already have a well established word for meaning m. Now the one agent, **A**, that doesn't know the word makes up its own word, w, for m. If the speaker also performs an update and **A** happens to be the speaker in a couple of games in a row in games where m is used, **A** will form a strong association between the "wrong" word w and m. At this point all other agents would be using a different word than **A**, and **A** would require multiple rounds as the hearer to align its lexicon properly with the others. Problems like this do not arise if the speaker doesn't perform updates.

When an agent is translating a meaning into a word, it simply selects the word with the highest score for the meaning. Similarly, a word is translated into a meaning by

choosing the meaning that has the highest associated score for the word. Given the lexicon in Table 1, for meaning $[0.5\text{-}1]_s$, the word *mamebo* would be chosen, because it has the highest score in the row. For the word *ranu*, the meaning $[0\text{-}0.5]_s$ would be chosen, because it has the highest score in the column.

2.6 Summary

The problem setting for language emergence in this thesis has now been described. This will serve as a starting point for the contributions of this thesis on grounding language in practicality beginning in the next section. Below a summary of the most important points to remember going forward.

- The agent society consists of autonomous agents learning individually through their own experiences (Section 2.1).
- The agents learn general, category-defining meanings that can in theory be applied to new environments directly (Section 2.3).
- The language emerges as a consequence of the agents repeatedly playing language games amongst each other (Section 2.4).
- An agent has a lexicon that maintains associations between meanings and words (Section 2.5.2).

3 Grounding language in practicality

The goal of this thesis is to ground language not only in perception, but the needs of the agents, i.e. practicality, as well. The needs of an agent arise from its goals, i.e. what it is trying to accomplish. However, the needs of an agent are not dictated by its goals alone, but also by the environment it is situated in. As what kind of a language is practical depends on the goals and the environment, the agents need some kind of abilities for guiding the language emergence according to their situation.

Practicality and its implications for language are discussed in more detail next. After that, the rest of the section is devoted for introducing abilities for practicality grounding.

3.1 Practicality

In this thesis a language is deemed practical, if it somehow helps an agent in reaching its goals. One could argue that creating or learning a language can be seen as a goal in itself, which is true. In fact, this has been the implied goal for the agents in previous research on language games in computational agent societies. This is because the research has focused on how a language might emerge and how different mechanisms affect the emergence (see Section 8 for more information). The agents have merely been vessels for the language, without any aspirations of their own. The performance of these agents has been mainly measured in terms of communication success rate, i.e. how often the agents succeed in the language games they play amongst each other.

The focus is shifted in this thesis by asking how a *practical* language might emerge. In other words, how might a language emerge, that the agents can use as a tool for performing their tasks. The agents don't develop a language just for the sake of developing a language, but as a means to an end. Communication success rate alone isn't a sufficient evaluation measurement anymore. It is possible for a society of agents to have a completely useless language with a perfect communication success rate.

Imagine for example the builder's language game described in Section 2.4, where a builder and an assistant had a language consisting of words for different kind of stones. The builder in this case is presumably trying to build something, which is why he needs the assistant to bring him the correct stones. The described language is great for this purpose, as the builder can use it to acquire the right stones. If instead the language consisted of names for different bird species, the builder would be in trouble, as this language couldn't be used to guide the assistant in the stone selection. Practicality of a language then seems to be dependent on its user's goals.

The builder's language game highlights another interesting aspect of practicality based grounding. The builder and assistant don't need to have the same goals in order for a language to be practical for both of them. As already mentioned, the builder's goal could conceivably be to build something. The assistant's goal could instead be just to keep his job, so that he can some day buy his dream car. Even though the builder and assistant have different goals, both benefit from speaking the same language.

An agent's goals, however, are not the only determining factor for what kind of a

language would be practical to it. Practicality is also dependent on the environment the agent is situated in. The builder and the assistant had the word "pillar". Let's say that in their world two types of pillars exist: type 1 and type 2. The word "pillar" can refer to both of these types. The builder only needs type 1 for the house it is building. If the construction site only had type 1 pillars, the agents wouldn't have any problem coping with just having the word "pillar" in their language. If the construction site had both type 1 and type 2 pillars, the assistant wouldn't know which kind of pillar to fetch, when he hears the builder shouting "pillar". A way to address the problem would be to have different words for type 1 and type 2 pillars. The only thing that changed between these two scenarios was the environment. The builder was still building the same house, and the assistant was still just trying to keep his job. The type 2 pillars weren't even used in the construction. Nevertheless the mere presence of them changed the requirements for the language.

3.2 Goal monitoring

An agent's ability to monitor how well it is reaching its goals is the key to grounding language in practicality in this thesis. More precisely, the agents in this thesis monitor how well they are performing a task. The agents use the information from the monitoring to learn meanings that affect their performance. If something doesn't affect an agent's performance, there is no need to communicate about it. If something does affect an agent's performance, it might be something worth conceptualising and communicating about. Of course if an agent is also interested in how other agents are doing, just considering one's own performance is not enough. But as already explained in Section 2.1, for simplicity's sake only agents that act solely based on their own goals are considered here.

How exactly the monitoring is used to derive practical meanings can be implemented in several ways. For instance, in the setting for the experiments conducted in this thesis, the agents make plans for moving from point A to point B. The plan in this case is the route which the agent is going to take. If everything goes to plan, the agent doesn't need to learn anything language related. In case something that hinders the plan happens, it is used as a learning opportunity. In the experiments, when an agent gets blocked by another agent in a tight space, the agent might have to backtrack and take a longer route than originally planned. A situation like this provides a great opportunity for playing a language game, because both agents are physically in the same place, their observations overlapping, while something

important is happening. Through language games they learn to refer to similar tight spaces. Then in the future, they can avoid being blocked again by communicating about the tight spaces they are planning to pass through. The experiment setting is detailed more in Section 4 and the language games are described in Section 6.

In addition to having practical meanings, it might be useful to model the importance of different meanings. In a situation where an agent can't communicate all the meanings that could possibly affect its plan, the agent has to choose what to communicate. If it can communicate only one meaning, then a natural choice would be to communicate the meaning that affects its performance most. The importance of a meaning then represents how big of an effect the meaning has to an agent's performance.

A sufficiently intelligent agent might be able to deduce the importance of a meaning in any given situation through some kind of a reasoning. A less intelligent agent however might have to resort to merely maintaining statistics on the importance of different meanings based on its experiences. These statistics could manifest themselves, e.g., as a scalar value associated with each meaning. Depending on whether the effect of a meaning is performance enhancing or hindering, the scalar value could be positive or negative. Continuing the example from above, when the agent gets blocked, it knows how much longer the new route is compared to the original plan. This information can be used to give an importance value to a meaning. If it only needs to take a couple of extra steps, the meaning representing that situation is not very important. If the new route is considerably longer, the importance value should reflect that.

Maintaining importance values allows the agents to avoid having to deal with functional meanings (see Section 2.3). An agent doesn't need to understand that it might need to backtrack and take extra steps if it is blocked in a tight space. All it needs to know is that a tight space is associated with a negative effect on its performance.

3.3 Determining context

For playing language games, two agents need a common context. In this thesis, the context is defined as described in Section 2.3, i.e. given the set of all objects O, the context is a subset $C \subset O$. Each agent has to determine the context individually. An agent can do that in two ways, detailed next.

First one is to use its sensors and determine the object set from what it is currently

perceiving. For example in a game with a speaker and a hearer, the speaker could simply look at a part of its environment with its camera. The context could be the set of objects inside the field of view of the camera, as in the Talking Heads experiment [11] described in Section 2.4. The hearer then would have to try to infer, which context the speaker has determined. This could be achieved by looking at the speaker and inferring where its camera is pointing at. Then the hearer could point its camera in the same direction. This method can be used when both agents can perceive the topic object of the language game.

Second one is to use its internal model, e.g. a map, of the environment. The agents can talk about objects in their models, such as parts of the map, that neither of them is currently perceiving. Because the agents can't rely on their perceptions to get a set of objects, they have to have some way of deriving objects from their internal model. Given the ability to detect objects through perception, as discussed above, an agent could simply keep track of the objects it has seen, and derive the context from them. Another approach, used in the experiments of this thesis, is to derive objects from the internal model of the environment using learned meanings. Suppose that the speaker has learned the meaning "corridor" and wants to talk about a "corridor" that is in its map. The context could be all "corridors" that are in the speaker's map. To give the hearer a hint of the context, the speaker could either point to a "corridor" both agents are perceiving, or say the word corresponding to "corridor" in their language. The hearer can then take all the "corridors" that are in its map as the context. To the author's knowledge, the context has not been determined in any previous research by deriving it from an internal model of the environment.

It is also conceivable to combine these two approaches in one language game. The speaker could describe an object that it is perceiving, while the hearer has to guess what kind of an object it is solely based on the description. Or the speaker could describe what kind of an object it wants the hearer to fetch. The hearer then would perceive a set of objects, possibly in another room, from which it has to choose the most suitable one.

In the ideal situation agents engaging in a language game would share the same idea of the context. In a game between agents A and B, that have determined the contexts C_A and C_B respectfully, this would mean $C_A = C_B$. In realistic scenarios it isn't reasonable to assume that both agents can derive precisely the same context. For instance in the Talking Heads experiment, the agents use cameras to perceive shapes on a white board. The context is determined by which objects are included

in the image captured by a camera. Each agent uses a different camera, so there is always a slight difference in the image they receive. Sometimes the difference causes the agents to perceive differing object sets, so that $C_A \neq C_B$. In the case the context is derived from an internal model, causes for different views of the context include differences in models between the agents and misunderstood communication about what the context should be.

In previous studies the context has been selected randomly. For practical purposes, the context has to be chosen so that the language games can convey information that is useful to the agents. In other words, playing language games has to allow the agents to gain knowledge they can use in their decision making. This can be achieved by playing games about objects that are important (as discussed in the previous section) and part of an agent's plan. For example, once an agent has planned a route for itself, it could start a language game where the topic of the game is the place with the most importance in the route. This place could correspond to a "corridor", which might have a negative importance value in case the agent has been blocked in "corridors" before. If the agent manages to indicate the "corridor" that is in its path from the context of all possible "corridors" to the other agents, the others would gain the knowledge of which "corridor" is part of the agent's plans. The knowledge could lead the other agents to avoid the "corridor".

3.4 Option-based discrimination

A common task in language games is to discriminate an object from a set of objects. For example, with a set consisting of a red object and two blue objects, the red object could be discriminated by saying "red". In precise terms, when an object o_j is to be discriminated given a set of objects $O = \{o_1, \ldots, o_i\}$, the goal is to choose a meaning m that only describes o_j, i.e. $O_m = \{o_j\}$. This was the task, for example, in the Talking Heads experiment [11].

In this thesis a slightly different discrimination problem is introduced. First of all, there is no single object to be discriminated. Instead, the agents discriminate a subset of objects, $D_1 \subset O$, from another subset of objects, $D_2 \subset O$. The subsets do not share any objects, i.e. $D_1 \cap D_2 = \emptyset$. The goal now is to choose a meaning m, so that $D_1 \subset O_m$ and $D_2 \cap O_m = \emptyset$. Note that this definition allows m to describe objects in O that are not in D_1, which means that O_m can be larger than D_1. The task could be generalised so that a subset is discriminated from an arbitrary number of other subsets, but in this thesis we only consider situations with two subsets D_1

and D_2.

The subsets D_1 and D_2 represent the options an agent has, which means that the goal of the discrimination is to distinguish an agent's options from each other. The motivation for this kind of *option-based discrimination* is that an agent needs information from communication, which helps it make decisions. If the content of the communication does not discriminate between an agent's options, it might not be very useful for decision making.

Suppose that the agents are in a 2D grid world, where they have to make choices over what route to take to their destination (this is actually the case in the experiment described in the next section). D_1 could be a set of cells in one route option and D_2 a set of cells in another route option. Depending on how the environment is laid out, finding a meaning that can perfectly discriminate D_1 from all other objects might be difficult, if not outright impossible. However, option-based discrimination doesn't require a perfect discrimination. For practicality a perfect discrimination is not necessary. The discrimination needs to only be accurate enough to allow the agents to make meaningful decisions.

In this thesis the agents use *option-based discrimination* to discriminate between different types of places on a map. The discrimination happens as a part of the Query Game, later detailed in Section 6.3. The implementation of the option-based discrimination is described in Section 6.3.2.

3.5 Summary

Grounding language in practicality has now been introduced on a conceptual level. To allow grounding in practicality, several abilities were introduced for the agents. These are the learning of meanings through goal monitoring (Section 3.2), determining the context of a language game from an agent's internal model using importance (Section 3.3) and distinguishing an agents options with option-based discrimination (Section 3.4).

It should be pointed out that none of these abilities are inherently required for grounding language in practicality, but are merely suggestions for doing so. Grounding language in practicality is an open-ended problem that can be approached with many kinds of methods.

This concludes the conceptual part of the thesis. In the rest of the thesis grounding language in practicality is investigated through empirical experiments conducted

through computer simulations.

4 The experiment setting

The experiment setting in this thesis revolves around a warehouse environment, where agents fetch items from shelves. The agents' goal is to be as efficient in this task as possible, which means that a language can be seen as practical if it increases the amount of deliveries the agents make.

The main purpose of this experiment is to see if the abilities for grounding language in practicality introduced in the previous section actually work. Simultaneously the implementation of the agents serves as an example for others who might want to create their own agents. Additionally, this experiment is used to learn about different aspects and challenges of practicality grounding through hands on experimentation.

The experiments are conducted in three distinct phases, each with its own dedicated section. The phases are shortly described next.

Phase 1 (Section 5) The basic environment and agent behaviour is implemented without language. The results from this phase serve as a baseline for the second phase, where a society utilising language is realised.

Phase 2 (Section 6) The purpose of the second phase is to set up an agent society, where a language grounded in practicality emerges. To this end two language games, the Place and the Query Game, are introduced and the abilities for grounding language in practicality discussed in the previous section are implemented.

Phase 3 (Section 7) In the third phase, different properties of the language are explored, such as how well the meanings learned in one environment transfer to another.

The rest of this section details the basic implementation of the experiment setup, that serves as a basis for all the phases. First, the warehouse environment and the agents and their behaviour is described. Then, the methods used for evaluating the experiments are discussed.

4.1 The warehouse environment

The agents reside in a warehouse, where their task is to fetch items from shelves. The environment is modelled as a two-dimensional grid, as depicted in Figure 2. A

grid cell can contain a wall, a shelf, an agent, or an order centre. A cell can't contain more than one object, making it so that the agents can't pass through non-empty cells. The order centre is where the agents get orders to fetch items. The items are also delivered to the order centre. Time in the environment passes in discrete steps.

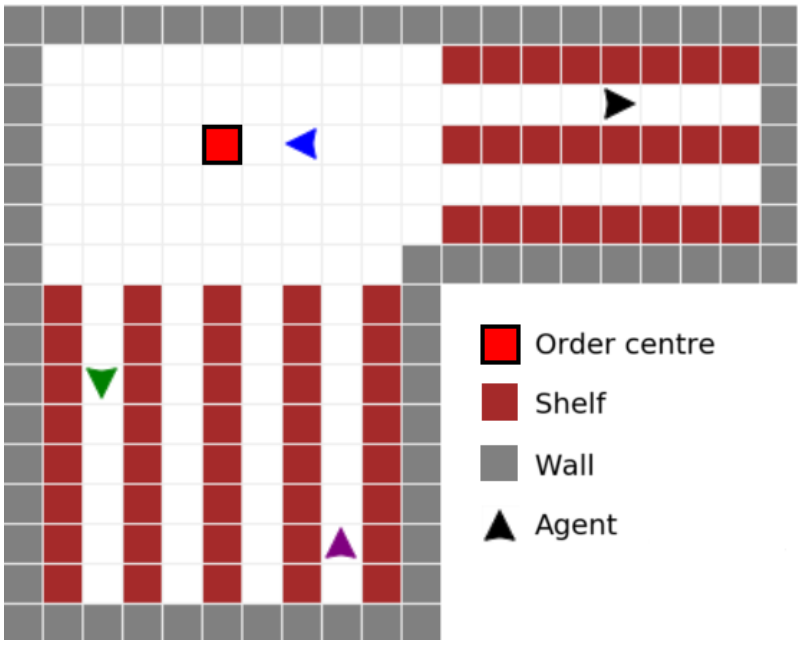


Figure 2: A warehouse environment with agents in example positions.

The environment has been designed with three aspects in mind. First, an agent should have a goal that it can monitor by itself. The goal of an agent is to deliver a specific item to the order centre. This task doesn't require an agent to be aware of which items the other agents are fetching. Also if something goes wrong during a delivery, i.e. the agent gets blocked and the delivery takes longer than originally planned, the agent can detect it.

Second, an agent should have options to choose from when acting in the environment. Language is used to select between these options. If an agent didn't have any options on how to act, language would be meaningless. The options in this environment arise from the fact that an item can be picked up from any side of a shelf. For example, when an item is requested from one of the shelves in the same column as the order centre in Figure 2, an agent can pick it up from either the left, or the right side of the shelf. Communicating with language can help the agent to choose which side it should go to.

Third, communication should be required for acting efficiently in the environment. In the warehouse, an agent's performance can be hindered only if it is blocked by another agent. The agents can act more efficiently, if they have some way of avoiding each other. The orders are given to the agents randomly, so that the agents only way of getting information about the location of other agents is through communication. If the orders weren't random, the agents could learn the pattern behind them, and use that knowledge to avoid each other. Of course learning a simple language might be faster, than learning the behaviour pattern of a whole society of agents, but investigating that is out of the scope of this thesis.

4.2 Agent

The agents have been designed for the experiment environment. They have been equipped with basic skills that allow them to perform deliveries even in the absence of language. Performance without language serves as a baseline to compare results once the agents utilise language. Every aspect of their behaviour is entirely predetermined, barred from the content of their communication. Next the basic behaviour without language is described. Then, how the agents plan routes is discussed.

4.2.1 Basic behaviour

The agents behave according to the following pattern:

1. Get an order (from the order centre) to fetch some item
2. Plan the route to the shelf that contains the item
3. Move to the shelf and pick up the item
4. Plan the route back to the action centre
5. Move to the action centre and drop the item
6. Start from 1 again

From this pattern follows, that an agent will be constantly moving back and forth between the order centre and a shelf.

During a single time step an agent can move one cell. All other actions such as picking up an item, playing a language game, planning a route, and turning do not take time.

A *collision* happens when an agent tries to move to a cell that already contains another agent. When two agents collide, the one that is not carrying an item will give way, unless it can immediately compute a new route. An agent gives way by taking random steps towards the action centre until its path to its destination is clear again. This behaviour has been designed to prevent agents from getting stuck in the environment.

4.2.2 Route planning

The route planning is done with the A* search algorithm [20]. To this end, each agent has a map of the environment, where each cell containing a permanent object (wall, shelf, or order centre), is marked. The location of another agent can also be marked on the map, if it is in an adjacent cell.

The environment has been designed in a way that in step 2 of the agent behaviour, when an agent plans a route to a shelf the agent has two options, i.e. which side of the shelf to pick up the item. How the agents utilise language to select between these options is described in Section 6. Here the agents have to make the choice without language and for this we test two alternative route planning schemes: *the shortest scheme* and *the random scheme*.

The shortest scheme An agent simply compute the shortest route to the shelf and uses that. Although using the shortest route seems intuitively very reasonable, it might perform poorly e.g. due to the agents crowding to the same places, which is why the random scheme is also tested.

The random scheme An agent first calculates the shortest route as in the shortest scheme. Then the agent computes a second route assuming that the last cell in the first route is blocked, giving the agent the second shortest route. Once language is implemented, the agents will utilise it to gain information over these two options to make better selections. Before the agents have language, they don't have useful information about the options and will make the selection between the two options randomly.

4.3 Evaluation methods

Before moving on to the experiment results, how the experiments should be evaluated needs to be discussed. The main measurement used is the amount of deliveries an agent makes during a simulation run, which reflects the performance of an agent. From the point of view of practicality it could be argued that this is the only measurement that matters, as practicality is only concerned with performance. Regardless, various other measurements are also taken to analyse the behaviour of the agent society and the emergence and properties of the language.

4.4 Summary

An experiment setting where agents fetch items from shelves in a warehouse has now been described. The agents are able to operate reasonably in this setting even without language, which offers a comparison point for once the agents utilise language. The agents' goal of making deliveries as efficiently as possible offers a way to measure the practicality of a language through the amount of deliveries the agents make.

5 Baseline experiment

To test whether the agents perform better with the shortest or the random scheme (see Section 4.2.2), experiments were run with 6 agents in the described warehouse environment, as depicted in Figure 2. For both schemes 100 simulations with 100 000 time steps were run. The reported values are averages from these simulations.

From Table 2 it can be seen that with the random scheme, an agent makes more deliveries than with shortest scheme. For this reason, random scheme will serve as the baseline for performance.

Measurement	Random scheme	Shortest scheme
Deliveries	$\mathbf{4414} \pm 5$	3668 ± 7
Collisions	$\mathbf{3670} \pm 13$	7067 ± 18

Table 2: Mean deliveries made and times collided for an agent during a single simulation run. The means are displayed with 99% confidence interval.

Although the agents take longer routes with the random scheme than with the shortest scheme, they make more deliveries, because they collide less. In Figure 3 it can be seen that with the shortest scheme the agents tend to clump around the bottom middle shelf, resulting in a lot of collisions. With random scheme there is less collisions and they are more spread out.

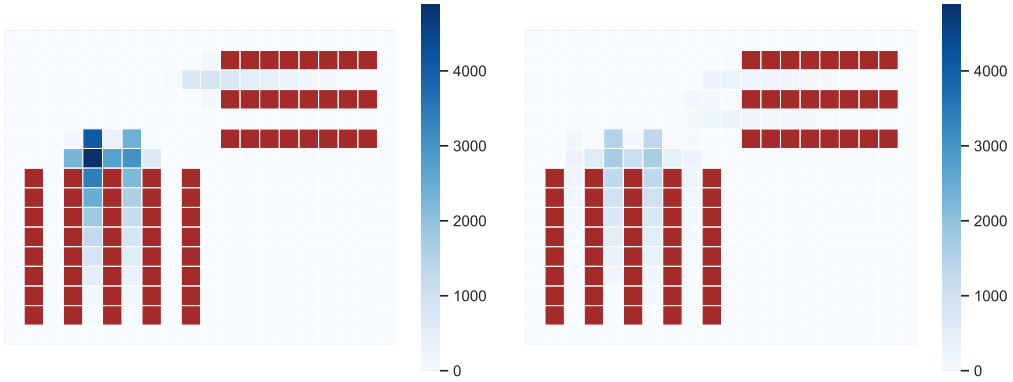


Figure 3: Collision heat maps for a society using shortest scheme (left) and a society using random scheme (right). Shelves are marked with brown squares.

6 Setting up language

The primary goal of the experiments is to show, that the abilities for grounding language in practicality introduced in Section 3 can work, i.e. result in a language that is useful to the agents. To this end these abilities are implemented in the agents here.

To facilitate the use of language two new language games are introduced: *the Place Game* and *the Query Game*. In the Place Game the agents learn words for different types of places in the environment. In the Query Game the agents learn words for discriminating between places of the same type. Additionally in the Query Game, the agents communicate useful information to each other that allows them to make better decisions. The Place Game can be played independently of the Query Game, but the Query Game requires the meanings learned in the Place Game. For this reason Place Game is described and tested first.

Next, the meanings the agents learn and communicate in the language games are detailed. Then the Place Game is described and experiments are conducted with it to see how the language emerges for different types of places. Finally the Query

Game is described and experiments are conducted in a society playing both of the games to investigate if the agents are able to utilise the emerging language in a practical way.

6.1 Meanings

In order to use language, the agents need to be able to learn meanings and associate them with words. The agents can learn two types of meanings: *place meanings* and *discrimination meanings*. Place meanings refer to different types of places on the map. Discrimination meanings are used to discriminate a place from other places of the same type. Using place and discrimination meanings together, the agents can pinpoint different locations on the map. How the meanings are learned and used depends on the language games the agents play.

A place meaning is a 3x3 grid, defining the contents of cells in a local area, as shown in Figure 4. An agent can perceive the 3x3 grid around it and memorise the grid as a meaning.

A discrimination meaning is a node in a discrimination tree, i.e. categoriser, as described in Section 2.5.1. The features used for the discrimination trees are the xy-coordinates of the cells in the map. An agent has a separate discrimination tree for both the x- and y-coordinate, allowing the agent to discriminate places based on location.

A place meaning can be used to define a subset of cells on the map, which corresponds to the context as discussed in Section 3.3. The set defined by a place meaning contains all the cells that correspond to it. A cell corresponds to a place meaning if the 3x3 grid surrounding it is equal to the 3x3 grid of the place meaning. See Figure 4 for an example.

Each place meaning is associated with an importance value (as discussed in Section 3.2). The importance value represents the effect the place has on the agent's performance. In the warehouse experiment setup, it translates into how many extra steps an agent has to take when a collision happens in a place corresponding to the place meaning in question. Of course in a complex environment, all places that correspond to the same place meaning probably don't have the same effect on performance, but modelling that is out of the scope of this thesis. Instead, all places of the same type are considered equally important.

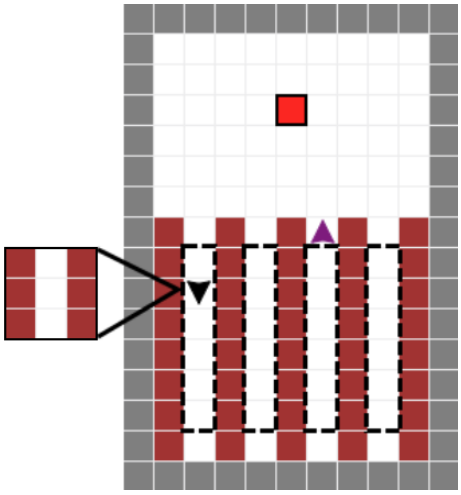


Figure 4: On the left the place meaning corresponding to the surroundings of the left agent. All the cells on the map corresponding to this meaning have been outlined.

6.2 The Place Game

The purpose of the Place Game is to allow the agents to learn and name place meanings. The Place Game is played between a speaker and a hearer. The speaker utters the word it has associated with its current surroundings. The speaker and the hearer update the association with its current surroundings and the word. An assumption is made in the Place Game, that the topic of the game is the place where the agents are currently situated. Alongside the Place Game, agents also update importance values for place meanings, although this is not directly part of the Place Game itself.

The script for the Place Game, with agent **A** as the speaker and **B** as the hearer goes as follows:

1. **A** perceives its current surroundings and derives a place meaning from it (the 3x3 grid around **A**).
2. **A** fetches the word associated with the place meaning from its lexicon. If there is no association, a new word is generated randomly, and it is associated with the place meaning.
3. **A** utters the word to **B**.
4. **B** perceives its current surroundings and derives a place meaning from it.
5. **B** updates its lexicon by incrementing the score between its derived place meaning and the uttered word. The score of all other pairs that contain either the meaning, or the word, is decremented by 0.1. (see Section 2.5.2 for details).

The Place Game is played when an agent that doesn't have an item tries to move into a cell that has another agent in it, i.e. when a collision happens. As was explained in Section 4.2.1, this will cause the agent to give way. At this point an agent can detect that its performance was hindered and it will initiate a Place Game. The agent giving way has to be the one to initiate the game, because the other agent has no way to detect that something important happened. The other agent will continue performing its task as if nothing happened.

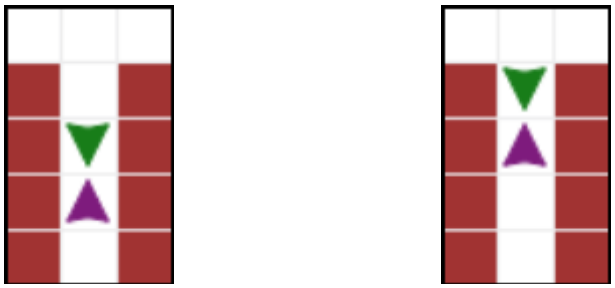
After the Place Game is played, the initiator agent will keep track of how many steps it has to take while giving way. The steps are counted until the route is clear again and the agent can continue its task. The amount of steps counted this way is noted with s. The place meaning, p, for which the importance value, i_p, is updated is the one the agent used in the Place Game. When an agent stops giving way, it will update i_p with the formula $i_p \leftarrow i_p + \lambda(-s - i_p)$, where λ controls the update rate. From the formula it can be seen, that the value of i_p is updated towards $-s$ (a negative value is used, because the effect is negative to performance). The result is an estimate of how many steps an agent has to take while giving way when a collision happens in a place corresponding to p. How λ should be picked depends on the variance of the variable that is learned. In this case, if the value of s varies greatly, a low value should be set for λ to prevent high oscillation of i_p. In case of low variance, λ can be set to a higher value to speed up learning. In this thesis λ is set to 0.1.

The formula was introduced by Claus and Boutilier [21] and is a simplified version of a common reinforcement learning method called Q-learning [22]. Originally the for-

mula is used to update so-called Q-values, that are associated with different actions an agent can take. A Q-value represents the estimated reward that is gained by taking an action. Here the formula has been adapted for the purposes of updating importance values associated with place meanings.

6.2.1 Example rounds

Figure 5 shows two example situations for the Place Game. An example of a game round is given for both next assuming that the speaker, **A**, is the top agent and the hearer, **B**, is the bottom agent.



(a) Symmetric situation (b) Asymmetric situation

Figure 5: Example situations for the Place Game.

Example 1, *symmetric situation*

The situation in Figure 5a is a *symmetric situation*, where the 3x3 grid around both agents is identical. Below is an example round in this situation:

1. **A** perceives the 3x3 grid around it and uses it as the place meaning.
2. **A** translates the meaning into the word *mamebo*.
3. **A** utters *mamebo*
4. **B** perceives the 3x3 grid around it.
5. **B** updates the association between *mamebo* and the 3x3 grid around it.

In this example **A** and **B** perceive the same 3x3 grid. **B** updates the association between the same meaning-word pair as was used by **A** and as a result **B** is more likely to use the same word as **A** for the same kind of a place in the future.

Example 2, *asymmetric situation*

The situation in Figure 5b is an *asymmetric situation*, which means that the agents have different kinds of 3x3 grids around them. As both agents perceive their surroundings independently for the Place Game, the agents will use different meanings in the game. Below is an example round for this situation:

1. **A** perceives the 3x3 grid around it and uses it as the place meaning.
2. **A** translates the meaning into the word *fomodu*.
3. **A** utters *fomodu*
4. **B** perceives the 3x3 grid around it.
5. **B** updates the association between *fomodu* and the 3x3 grid around it.

This time **B** had a different perception of the surroundings than **A**. **B** updates the wrong meaning-word pair in its lexicon, making misunderstandings between the agents more likely in the future.

6.2.2 Experiments

Experiments were run in a society where the agents play The Place Game and learn importance values. The experiment setup is exactly the same as the previous setup in Section 5, except with the Place Game. The goal of these experiments was to validate that a language for places emerges and that the agents can learn meaningful importance values for the place meanings. The performance of the agents is not evaluated here, because playing the Place Game does not affect the agents route choices or anything else that could affect their performance compared to the previous experiment setup. 100 simulations were run with 100 000 time steps and 6 agents. The run averages are reported. The simulations were run in the same warehouse environment as the experiments without language in Section 5.

A direct comparison of the agents' lexicons reveals that they are highly incoherent. By the end of a simulation, only 68% of all learned meanings are in every agent's lexicon. Out of the meanings learned by all agents, only 14% are translated to the same word by every agent. The cause for such incoherence is the way the agents have been implemented. They can't form a language for asymmetric situations where the

speaker and the hearer have different observations of the environment (see Example 2 in previous section). Of the games played only 35% are in symmetric situations.

Figure 6 shows that overall in all games, the agents have a low success rate. A game is successful if the hearer's interpretation of the uttered word is correct, i.e. it matches the meaning meant by the speaker. [3] Otherwise the game fails. However, when only symmetric games are considered, the success rate approaches 100%.

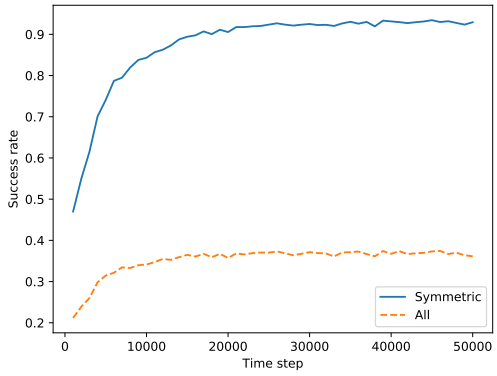


Figure 6: The success rate of games played smoothed over 1000 time steps. Showing only 50 000 time steps, after which the curves plateau.

Figure 7 confirms that maintaining the importance values works reasonably. Higher importance is given to places where there is a higher chance of having to back away for longer time periods.

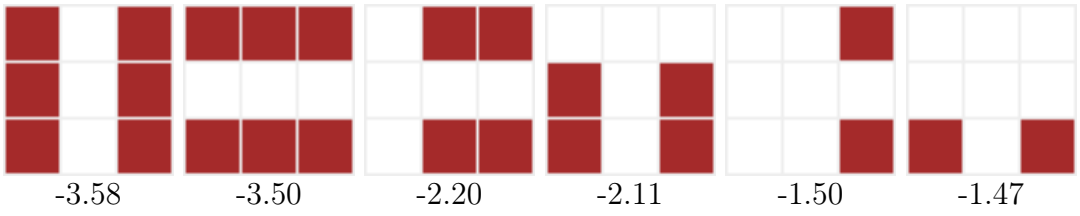


Figure 7: Importance value for the six most important places.

For the top two most important meanings the language is highly coherent. For the most important meaning all agents use the same word by the end of the run. For the second most meaning all agents use the same word by the end of 93% of runs.

[3] The interpretation is not used in the game itself, and information of the success is not available to the agents. They are only used for analysis purposes.

6.2.3 Discussion

Although overall the learned language is incoherent, it doesn't mean that the language can't be practical. In the experiment settings of this thesis, the language needs to be coherent only for the most important meanings. This is because they are the ones used in the Query Game, which the agents use for selecting routes. Indeed, for the two most important meanings a coherent language emerges. In other settings a more coherent language might be required for less important meanings as well.

The incoherence of the language is caused by how the perception of the place meaning is implemented. Because the agents only consider the 3x3 grid around themselves, there are several situations where perceptions differ. To get around this problem the agents could be implemented in multiple different ways. For example, the agents could perceive the 3x4 grid around both of them or they could use perspective alignment [23] to understand each others visual perspective.

To conclude it seems that the current implementation of the agents and the Place Game are good enough for the purposes of this thesis. The emerged language is coherent for the most important place meanings that are used for determining the context (see Section 3.3) for the Query Game.

6.3 The Query Game

The Query Game has two goals. First, the agents learn and name discrimination meanings. Second, it should give the agents information that allows the agents to make decisions about how they should act in the world. The Query Game is inspired by the variation of the Guessing Game that was used in the Talking Heads experiment [11]. For brevity, Guessing Game refers to this variation in the rest of this section. The Guessing Game is designed so that a language for discriminating shapes on a whiteboard emerges in the agent society. Likewise, the Query Game is designed so that a language can emerge for discriminating between places in the environment. In both games the discrimination is based on discrimination trees. The trees are grown and words for nodes in the tree (categorisers) are learned by playing the game repeatedly. [4]

[4] What objects are discriminated and how the discrimination is done is implementation-specific. Both games can be implemented to discriminate different kinds of objects and using other categorisation methods than the discrimination tree.

Part of the framework provided by a language game is an assumption about what the communication is about. In the Guessing Game it is assumed that a word is used to discriminate an object from all other objects in the context of a game round. In the Query Game the words are assumed to refer to a route that an agent is planning to take. As a result of this assumption the agents get information from the Query Game they can use to reason how well their route plans fit together.

The Query Game gets its name from how it is played. When an agent gets an order to fetch an item, it will plan a route to the shelf that contains it (2. step of the basic behaviour, Section 4.2.1). To determine whether a route is free, the agent will make a query to all the other agents by playing the Query Game. First, it will select a place meaning and a discrimination meaning that distinguish the route. The meanings are translated into words, which are broadcast to the other agents. The other agents interpret the words and guess what place is meant by them. If any of the other agents think that it is in the place, or that the place is on its route, it can *object*. If any agent objects, the querying agent will conclude that the route is not free. Otherwise, the route is assumed to be free.

As an alternative to the querying method, a solution was considered where an agent would privately decide its route and then broadcast its decision. The drawback of this is that the agents get the information they need for decision making spread over time. Therefore they have to keep track of what each other agent has broadcast. The query method was designed to give the agents all the information they need at the time of the decision, so that they wouldn't have to maintain models about which places are or aren't free.

The practicality of the information conveyed in the Query Game is dependent on which place and discrimination meanings are chosen. The place meaning is used to determine the context of the game. Given the context, the discrimination meaning is used to discriminate cells that are on the route from the other cells in the context. For selecting and using the meanings new methods are introduced. Selecting and using the place meaning is discussed in the next section. After that, selecting and using the discrimination meaning is discussed. For now it suffices to know, that the meanings are chosen in a way, that allows the agents to communicate about a route.

The Query Game is played in two parts. Part 1 is used to learn discrimination meanings, and for choosing a route to an agents destination. Part 2 is used to update lexicons. Below is the script for part 1 with **A** as the speaker. Before the game starts **A** calculates the two shortest routes to its destination (see Section 4.2.2

for full details). The subject of the first query made is the shortest of these two routes.

1. **A** selects a place meaning and a discrimination meaning for the route. If no place meaning exists on the route, the game ends. If no suitable discrimination meaning exists, the game ends, and the agent grows one of its discrimination trees.

2. **A** translates the meanings to words and broadcasts them to all other agents. If no word exist for the discrimination meaning, one is generated randomly.

3. The other agents interpret the words and guess which cells on the map they refer to.

4. If an agent thinks that at least one of the cells is on its current route, it will object.

If no one objected, **A** will assume that the route was free and chooses it. If someone objected, **A** will make another query, where the subject is the second, longer route. If no one objects, the second route is chosen. Otherwise one of the two routes is chosen randomly. In case the other agents don't know the words that were broadcast, they will simply stay silent.

Part 2 is played when two agents collide. As was with the Place Game, the agent that has to give way initialises the game, because only it detects that something went wrong. The agent initialises the game only in the case that it is currently in a cell, that corresponds to the place and discrimination meanings it used the last time it acted as the speaker in part 1 of the game. In other words, a collision happened even though this place was communicated about. There are two reasons why this might happen. First, the communication might have been misunderstood. Second, neither route considered by the speaker was free. In either case, this is a good opportunity for the agents to align their lexicons. The script for the second part with **A** as the speaker and **B** as the hearer is as follows:

1. **A** utters the word for the discrimination meaning it used in part 1.
2. **B** checks whether the cell it is in corresponds to the place and discrimination meanings it used the last time it acted as the speaker. If not the game ends.
3. **B** updates the score for the association between the discrimination meaning it last used acting as the speaker in part 1 and the uttered word (see Section 2.5.2 for details).

B has to check if the place corresponds to the last place it discriminated, because it has to ensure that the updated discrimination meaning is the one it uses to discriminate its current place on the map. Alternatively the agents could keep track of which discrimination meanings they have used for different parts of the environment, but for simplicity's sake the former option was used.

6.3.1 Selecting the place meaning

The selected place meaning is used for deriving the context for the Query Game from an agent's internal model, or map, of the environment. This is required, because in the Query Game the agents communicate about their future plans. More precisely, the agents communicate about places where they are going to be at some point in the future and therefore might not be currently perceiving these places.

For determining the context, the agents use place meanings and importance values learned from the Place Game. When an agent makes a query on a route, for each cell in the route it checks which place meaning the cell corresponds to and what the importance value of that place meaning is. The place meaning with the highest absolute importance value is chosen to determine the context for the Query Game. Absolute value is used, because an importance value can be positive or negative. In the experiments of this thesis the agents only learn negative values. Choosing the place meaning this way corresponds to the agent asking itself what kind of a place on my route affects my performance the most.

Once the place meaning has been chosen, all the cells corresponding to it are collected from the map. These cells are the objects that constitute the context. Within the context, each cell's x- and y- coordinates are normalised, so that relative meanings can be used (as discussed in Section 2.3).

6.3.2 Selecting the discrimination meaning

After the context has been determined, a discrimination meaning is chosen to discriminate the route in question. The topic of the game is the intersection of cells on the route and cells in the context. As a result, the topic can be a set of multiple objects, instead of just one object (as discussed in Section 3.4). The goal of discrimination is to distinguish the topic cells from the other cells in the context.

The discrimination meaning doesn't necessarily distinguish the topic cells from the other cells perfectly. For example in Figure 8, a perfect discrimination would be impossible for either of the depicted routes, given that only one coordinate can be used. In the implementation described here, the result of discrimination will be a set of cells, that contains at least the whole topic set. In precise terms, the result set is a superset of the topic set.

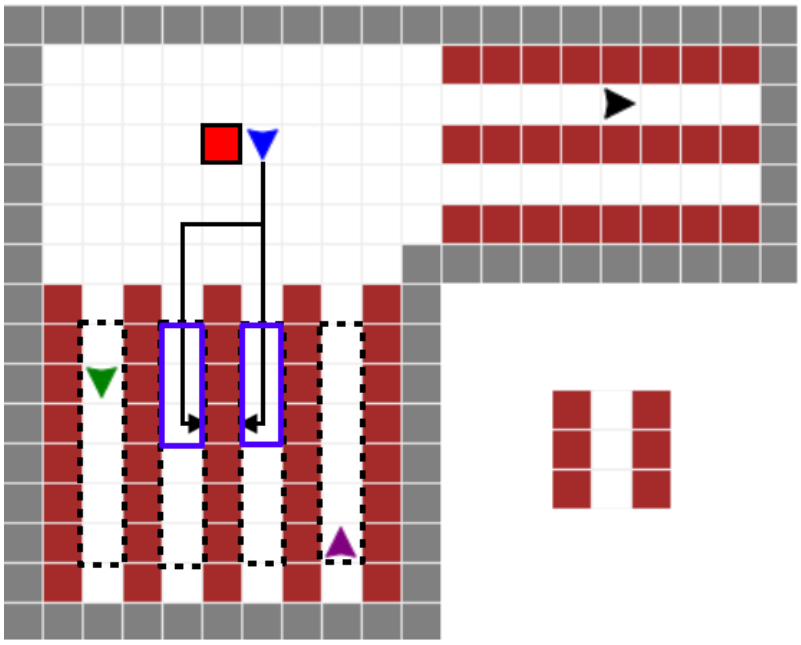


Figure 8: Two shortest routes shown for the agent next to the order centre. The place meaning used to define context is shown on the bottom right. The context comprises of the cells outlined with a dotted line. The topic cells have been outlined with a solid line for both routes.

To make a discrimination, an agent has to choose which feature to use and which level of the discrimination tree to use. The level of the tree can be thought of as

the accuracy of the discrimination. To illustrate, the cells in the context of Figure 8 have the following x-coordinates when normalised: 0.0, 0.3, 0.7, 1.0 (values have been rounded). The cells in the longer route on the left, outlined with blue have x-coordinate 0.3. The value falls in to the range of $[0\text{-}0.5]_c$ and its child $[0.25\text{-}0.5]_c$. Even further, due to how the discrimination tree is structured, an arbitrary number of levels can be added to the tree that all would have a categoriser containing 0.3. The further down the tree is travelled, the smaller, more accurate area is defined around the value, possibly defining a smaller result set. Using the categoriser $[0\text{-}0.5]_c$ would define a result set with all the cells, whose x-coordinate is 0 or 0.3. Categoriser $[0.25\text{-}0.5]_c$ would define a smaller set, with only the cells, whose x-coordinate is 0.3.

In the Guessing Game the level is chosen by travelling down the tree and selecting the first categoriser that only contains the topic object. Thus, the least accurate level is chosen that can unambiguously discriminate the object. In the Query Game this method has been modified to deal with a result set that can contain non-topic objects. The categoriser is chosen, that makes the result set smallest (and still contains the whole topic set). If several such categorisers exist, the one with the largest range is chosen. In other words, the agents use the least accurate categoriser that defines the smallest result set. See Figure 9 for an example of the discrimination process.

The feature selection is solved in the Guessing Game by gathering the categorisers for each feature, that can unambiguously discriminate the topic. Then the categoriser with the highest score is selected. In another manifestation of the Talking Heads Experiment, an alternative method is used to select a feature utilising saliency [5]. Saliency is defined as the distance to the closest value in the context. So if f_i is the value of a feature for object i and C is the context, saliency is $\arg\min_j \text{distance}(f_i, f_j)$, where $j \in C, j \neq i$. The most salient feature is selected.

Neither of these ways to select the feature is applicable in the warehouse scenario, because the practicality of the discrimination is dependent on the current situation, not previous success or saliency. Instead, *option-based discrimination* is used (see Section 3.4). The goal of the Query Game is to help the agent to make a choice between two options, which are the two shortest routes. Therefore, the agent should perform the discrimination in a way, that distinguishes the routes from from each other. A discrimination that doesn't distinguish the two routes would be less helpful in decision making.

Let's examine the longer route in Figure 8. The cells outlined by blue on the left

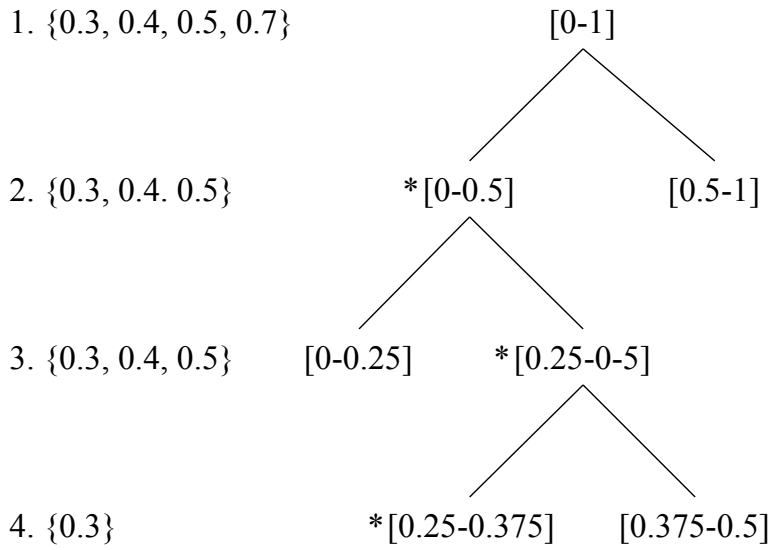


Figure 9: The discrimination process shown for topic set $\{0.3\}$. The result set for each level is shown on the left, containing the whole context on the first level. Traveling down the tree, a categoriser that contains the whole topic set is chosen (marked with an asterix). With topic set $\{0.3\}$ the categoriser on level 4 is chosen, as it achieves an unambiguous discrimination. With topic set $\{0.3, 0.4\}$, the categoriser on level 2 would be chosen, because there are no categorisers below it that contain the topic set and make the result set smaller. In this case, a perfect discrimination can't be achieved and the final result set contains a value that isn't in the topic set.

can be discriminated using the first level of the discrimination tree for both x- or y-coordinate. For x-coordinate, the categoriser would be $[0\text{-}0.5]_x$ and for y-coordinate $[0.5\text{-}1]_y$. Both of the categorisers would define a result of the same size. [5] However, only the x-coordinate distinguishes the route's from each other and is therefore selected.

In case an agent doesn't have a categoriser that can discriminate the routes in the first step of part 1 of the Query Game, it will grow one of its discrimination trees. The agent gathers all the leaf nodes from all of its trees and it checks if some leaf node could be split to acquire a categoriser that can discriminate the routes. If such a leaf is found, it is grown. Otherwise a random leaf is grown.

[5] Of course in this case the x-coordinate could be categorised more accurately, but this is not the case in all situations, and the agent might not have grown the tree that far.

The methods introduced above for feature and level selection are just one possible implementation. They have been designed so that in most cases agents with similar behaviour will discriminate the same places similarly. Therefore, when two agents collide and play the second part of the Query Game, they are likely to use the same discrimination meaning. Consequently the agents can use a simple lexicon, where they only update the score of one meaning-word pair per game. If the agents would regularly discriminate the same place in different ways, either the agents would have to be able to reason about how the others discriminate, or the lexicon (or its update scheme) would have to be changed to deal with it.

The discrimination method proposed here rests on the assumption that discriminating options from each other will eventually refine the discrimination accuracy to a sufficient level. This holds true in the warehouse environment, but might not always be the case. Imagine for example an environment with four "corridors" in a row. One of an agent's options is always either the first or second "corridor" from the left. The other one is always either the first or second "corridor" from the right. Hence, the options can always be discriminated from each other with the second level of a discrimination tree, i.e. with "left" and "right". An agent will never develop more accurate concepts like "very left" and "slightly left", although they might provide a more practical level of communication. Possible solutions for this are learning the correct discrimination accuracy through experience, or having the ability to reason about what level of accuracy is practical.

6.3.3 Example rounds

The first two examples are of part 1 of the Query Game, where the agent in Figure 10 is making a query. This agent is noted with **A**.

Example 1, part 1 success

1. **A** selects the place and discrimination meanings as seen in Figure 10.
2. **A** translates the place meaning into *mamebo* and discrimination meaning into *fomodu* and broadcasts them.
3. All other agents interpret the words correctly.
4. The agent directly below the querying agent sees that its path crosses the cells defined by the meanings and answers no.

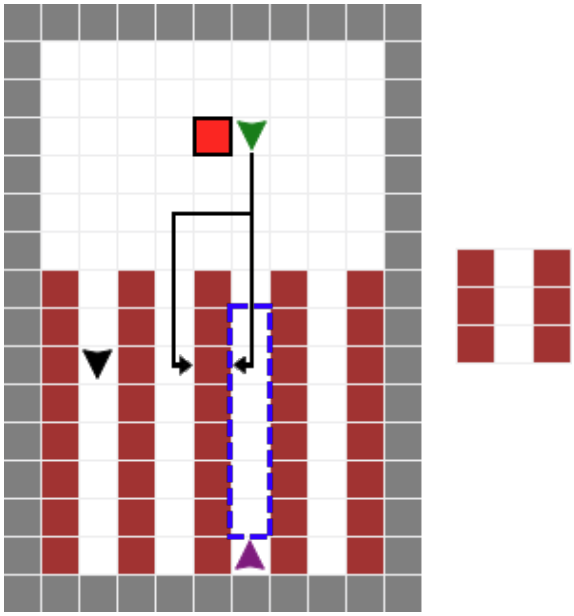


Figure 10: An example situation where the agent next to the order centre is making a query about the route on the right. The cells defined by the place meaning shown on the right and categoriser $[0.5\text{-}0.75]_x$ are outlined with a dotted line.

As an agent answered no to the query, the querying agent will next make a query for the left route in the same way. If the other agents interpret the query correctly again, they won't answer and the querying agent will choose the left route.

Example 2, part 1 failure

1. **A** selects the place and discrimination meanings as seen in Figure 10.
2. **A** translates the place meaning into *mamebo* and the discrimination meaning into *fomodu* and broadcasts them.
3. The agent directly below the querying agent misinterprets the words.
4. No agent answers the query.

Now the querying agent assumes that the route is free and will select it. This leads to a collision as seen in Figure 11. An example of a round of the part 2 of the Query Game played as a consequence of the collision is given next. **A** is the agent that made the query and **B** is the other agent.

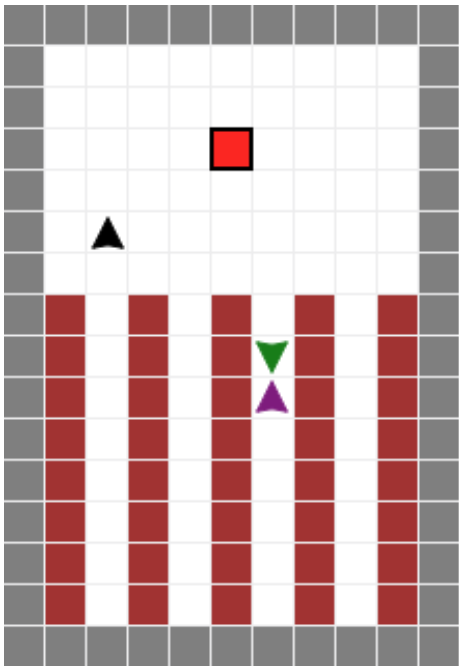


Figure 11: An example situation with a collision.

Example 3, part 2

1. **A** utters *fomodu*, which was the word for the discrimination meaning in part 1.

2. **B** sees that it is in a cell currently that corresponds to the meanings it last used as the speaker in part 1.

3. **B** updates the association between the word **A** uttered and the discrimination it used the last time.

There are two possibilities for the discrimination meaning **B** updates in this round, which are $[0.5\text{-}1]_x$ and $[0.5\text{-}0.75]_x$. The longer the simulation goes on, the higher the chance is that an agent uses the latter discrimination meaning, because it is accurate enough to discriminate between the two rightmost "corridors" between the shelves.

6.3.4 Experiments

These experiments were run with an agent society playing both the Place Game and the Query Game. The main goal was to see if utilising the language for route selection helps the agents deliver more items, i.e. perform better, compared to

the basic behaviour with the random route selection scheme and no language. 100 simulations with 100 000 time steps and 6 agents were run. The reported values are simulation run averages. The same warehouse environment was used as in the previous experiments in Section 5 and Section 6.2.2.

Performance An increase in performance can be observed in Table 3, as the agents start utilising language by playing the Query Game. There is an 9.3% increase in deliveries made with language compared to no communication, showing that the agents can utilise language in a practical way.

Measurement	Random scheme	Language
Deliveries	4414 ± 5	$\mathbf{4825} \pm 6$
Collisions	3670 ± 13	$\mathbf{2219} \pm 15$

Table 3: Average deliveries made and times collided for an agent during a single simulation run. The random scheme values are from Section 5.

Coherency For the x-axis discrimination meanings $[0\text{-}0.25]_x$, $[0.25\text{-}0.5]_x$, $[0.5\text{-}0.75]_x$, and $[0.75\text{-}1]_x$ all agents use the same word by the end of all runs. The same goes for y-axis discrimination meanings $[0\text{-}0.5]_y$, $[0.5\text{-}1]_y$. A language emerges for these discrimination meanings, because the agents use them in 99.0% of cases in the given warehouse environment. The agents use more accurate discrimination meanings for x, because there are more vertical than horizontal shelf rows. For other discrimination meanings a coherent language doesn't emerge.

It is important to note that the amount of agents translating a meaning to the same word doesn't tell the whole story of the coherence of the language. Because the lexicon supports synonymy, agents using a different word might still understand each other. In this experiment, however, synonymy seems to be rare. Only in 0.6% of the times when a hearer interpreted the speaker's word for a discrimination meaning correctly, the hearer would have used a different word than the speaker.

To measure the success of the agents' communication, two definitions are made for when a game is successful. The definitions are called *partial* and *complete* success. When an agent makes a query, i.e. plays part 1 of the Query Game, it broadcasts two words to the other agents. The game is partially successful if at least one of the hearers interprets the two words as the meanings intended by the speaker. Complete success is reached if all hearers interpret the words as the intended meanings. Only

part 1 of the Query Game is considered, because that is when the agents communicate and interpret the information needed to make decisions. Therefore the agents' performance is directly tied to how well they understand each other in part 1.

Performance w.r.t. language emergence In Figure 12 it can be seen that the average delivery time decreases as the language emerges. Interestingly, most of the decrease in delivery time has already happened when complete success is only around 30%. This indicates that a language can become practical early in its development, even if it is incoherent in the society as a whole.

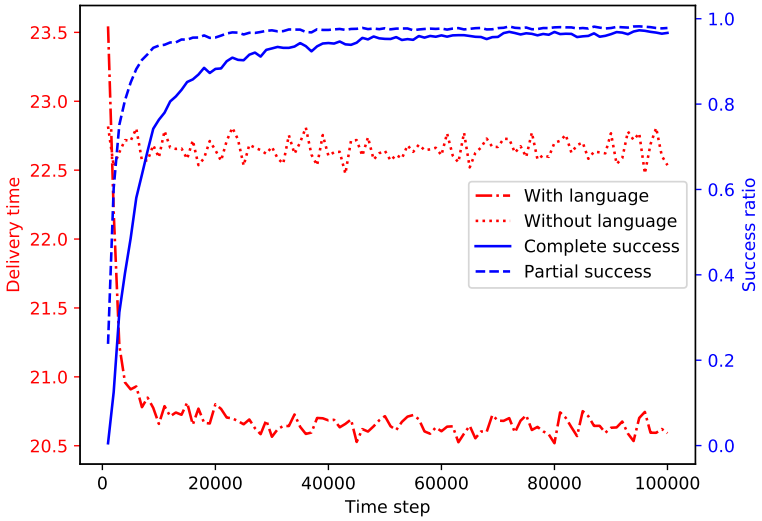


Figure 12: Showing the average delivery time of deliveries made by an agent with and without language. Also showing the complete and partial success ratios of language games played. All curves are smoothed over the last 1000 time steps.

Emergence speed Figure 13 gives an idea of how fast a coherent language emerges w.r.t. game rounds played. For partial success, an 80% success ratio is reached around around 1300 games. For complete success 80% is reached around 4800 games.

Environment's effect How much the language affects performance is largely dictated by the environment. To investigate this a 100 simulations were run, where the shelf rows were extended to be double the length of the default layout. Otherwise this extended setup was identical to the described Query Game setup. In the extended setup, there is a larger punishment for a collision between the shelves, because an agent has to back away longer on average, than in the Query Game setup. The relative increase in deliveries made is larger for the extended, being

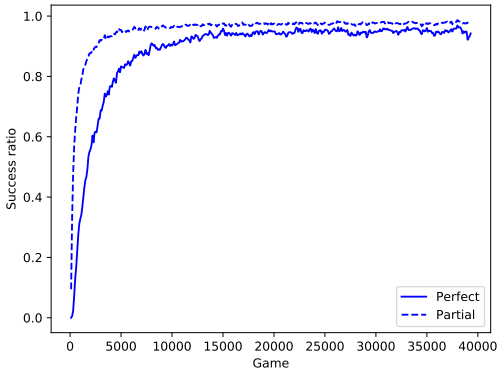


Figure 13: Showing partial and complete success ratios w.r.t games played. The ratios are smoothed over the last 100 games played.

10.4%, compared to the 9.3% for Query Game setup mentioned above.

6.3.5 Discussion

The 9.3% increase in deliveries made with language might seem quite small given the amount of work that went into designing and implementing the agents' abilities and the language games. Of course this increase is only specific to the presented implementation of the warehouse and the agents might benefit from language more in other settings as was seen in the experiment with extended shelf rows.

Unfortunately the language emerges slowly even in the simple simulation environment, which might limit the possible real life applications of the presented methods. However, no downside was detected to using language in the early learning phase and the language started benefiting the society quite early in its development. Also the methods used for language emergence here weren't really designed with the optimal emergence speed in mind, so there are probably many ways in which the emergence could be sped up.

6.4 Summary

The language games and agent abilities used for language emergence have now been implemented and the resulting language was shown to be practical. By playing the Place game the agents learn a language for different types of places and with goal monitoring the importance of these place types. Utilising the importance values

learned alongside the Place Game the agents were able to determine the context for the Query game in a useful way. Furthermore, with option-based discrimination the agents learned to discriminate places in a meaningful way.

7 Experiments on the language

Now that a society where a practical language can emerge has been implemented, it is easy to create new experiments exploring different aspects of the language and its emergence. This section consists of three such experiments, starting with an experiment exploring how the language could be used to learn to conceptualise route options. In the second experiment the language is tested against a reasonable predetermined behaviour that doesn't utilise language, but still manages to eliminate collisions. The third experiment explores if the agents can transfer their discrimination meanings to a new environment.

7.1 Route conceptualisation experiment

An essential part of the agents' behaviour is computing the two shortest routes, as they provide the options that the agents have a choice over. In the experiments above, the agents had a predetermined way of doing this. First they computed the shortest route. Then they computed the second route by assuming that the last cell in the first route is blocked. Although this way of computing the routes worked in the default warehouse environment, it doesn't translate well to other kinds of environments. In this section an alternative method is investigated that allows an agent to learn to conceptualise routes based on its experiences. The new method takes advantage of the agent's already learned importance values and lexicon.

For these experiments the agents are moved to a new environment depicted in Figure 14. This new warehouse also stores beer, which the agents are not allowed to handle. The agents operate as before, fetching items from the shelves. As the shelves are now placed in a way that leaves open room in front of them, the old way of calculating the routes does not produce meaningful options anymore. Also the places where collisions are most harmful are now in the middle of the route, instead of at the end. In the environment there are choices of route also when returning from a shelf to the action centre. The agents utilise the Query Game for this choice as well.

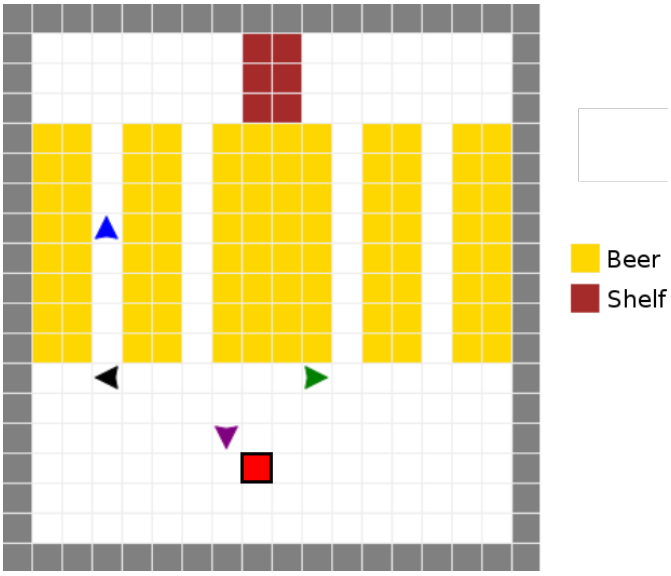


Figure 14: A warehouse environment that stores beer.

The new route conceptualisation method works as follows:

1. The shortest route is computed.
2. The most important place from this route is determined using the importance values.
3. The second route is calculated assuming that the cells corresponding to the most important place on the shortest route are blocked.

This route conceptualisation method is closely tied to how the Query Game is implemented. In the Query Game the agents discriminate a route based on the most important place meaning on it. Therefore the agents use the same place meaning to both conceptualise and communicate about their routes.

Using only importance values for the route conceptualisation might lead to situations where the agents use meanings they can't communicate to others. Thus *score filtering* is introduced, which is a method that allows the emerged language to affect what meanings the agents use for the route conceptualisation. When an agent is determining the most important meaning on a route, it will only use meanings that have a high score in the lexicon. More precisely, a meaning is only considered if

$s \geq F$, where s is the highest score of all meaning-word pairs containing the meaning in question and F is the filtering threshold. If $s < F$, the meaning is not used. In this thesis $F = 0.5$, which means that a meaning is used for route conceptualisation if an agents hears the same word for the meaning in 50% of Place Games. Preliminary tests indicated that this value was best for performance out of values $0.1, 0.2 \ldots 1.0$ in the experiment environment.

Again, the whole system is not implemented right away. First, agents without route conceptualisation or language are tested to set a baseline. Second, the route conceptualisation is implemented and tested without language games to investigate if route conceptualisation alone is already beneficial to the agents. Finally, route conceptualisation is used and language games are played to see how well the route conceptualisation works together with language.

An agent without route conceptualisation behaves differently when a collision happens than previously described. Above the agents gave way by taking random steps towards the action centre until the chosen route was clear again. An agent can't behave this way, unless it has a concept of what its route is. Instead an agent without route conceptualisation will calculate a new route when it collides without any constriction on having to return to the old route. If a new route can't be calculated, it will give way until any route to the destination can be found again.

The experiments are divided into post- and pre-language setups. With the post-language setup, the route conceptualisation is tested before the agents utilise language and with the pre-language setup, utilising route conceptualisation together with language is tested.

As before all simulations settings are run 100 times, and the run averages are reported. Each simulation run consists of 100 000 time steps and has 4 agents.

7.1.1 Pre-language setup results

The aim of this experiment setup is to analyse how the agents conceptualise their routes and to find out whether route conceptualisation is already beneficial to the agents before language is used. To this end simulations are run with agents with and without route conceptualisation. With route conceptualisation the agents choose their route randomly between the two shortest options.

It is observed that the implemented route conceptualisation doesn't benefit the agents without language. The average amount of deliveries an agent makes without

route conceptualisation is 2322 ± 2 and with route conceptualisation 2300 ± 3. The values are presented with 99% confidence intervals.

Figure 15 shows how the agents tend to conceptualise routes. The left meaning is used 85% and the right 7% of the time. These two meanings are the top two most important place meanings, because the agents might end up backing through the whole "corridor" if a collision happens in one of them.

Figure 15: Two most used meanings to conceptualise a route without score filtering.

7.1.2 Post-language setup results

In this setup the agents play the Place and Query games and use route conceptualisation. These experiments have two goals. First, to test if the route conceptualisation works together with language. Second, to see whether the agents can use score filtering to alter their route conceptualisation in such a way that results in more deliveries. The delivery amounts are presented with 99% confidence intervals.

Without score filtering an agent makes 2453 ± 8 deliveries, which is a 5.5% improvement over the pre-language setup. However, the most used meaning for route conceptualisation is the first one in Figure 15. This meaning is learned in asymmetric situations where the speaker's and hearer's perception of the place differs and therefore can't have a coherent word in the agent society.

With score filtering an agent makes 2531 ± 2 deliveries, improving the result without score filtering by 3.2%. The total improvement over the pre-language setup is 9.0%. With score filtering the most used route conceptualisation meaning is the right one in Figure 15. For this meaning all agents use the same word by the end of all runs.

The difference in the conceptualisation meanings used with and without score filtering is reflected in the complete success ratio shown in Figure 16. Because there is no coherent language for the most used place meaning without score filtering, the success ratio stays low the whole run. With score filtering the agents are able to communicate successfully.

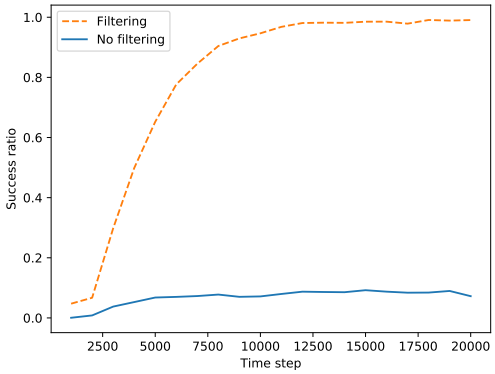


Figure 16: Complete success ratio with and without filtering. Only showing the first 20 000 time steps, after which the curves stagnate.

7.1.3 Discussion

Two new abilities, route conceptualisation and score filtering, were introduced for the agents. They were successfully implemented in the agents, leading to improvements in performance.

Although route conceptualisation alone without language didn't prove to be beneficial in the experiment environment presented here, it is conceivable that the outcome could be different depending on the environment. Also route conceptualisation could be generalised to option conceptualisation, meaning the agents could learn to conceptualise all kinds of options for themselves through experience, not just different routes.

An interesting interaction was seen in these experiments between language and how the agents conceptualise the world. It seems logical, that if the agents communicate about the world, they need to conceptualise it in a way that can be communicated. In other words limitations in the language impose requirements on the world conceptualisation.

7.2 Circling behaviour experiment

The goal of these experiments is to investigate the agents' performance against a predetermined behaviour designed to minimise collisions. The same environment from the above section is used (see Figure 14). With the predetermined behaviour, when an agent goes to a shelf, it will always use the outer "corridor". So if the

shelf is on the left, the agent uses the left "corridor", and when the shelf is on the right, the right "corridor" is used. When an agent returns from the shelf to the action centre, it will use the inner "corridor". With this behaviour collisions in the "corridors" are eliminated, because all the agents move in the same direction in all of them. As the behaviour results in the agents going in circles, it is referred to as *circling behaviour*. As the routes are predetermined with circling behaviour, the agents do not use language.

Simulations were run for 2, 3, 4, and 5 agents. With each amount of agents 100 simulations with 100 000 time steps were run. The reported values are run averages.

7.2.1 Results

In Figure 17 it can be observed that if the amount of agents is 3 or less, the language using agents perform better than agents with circling behaviour. However, with 4 or more agents circling behaviour has better performance. It seems that there is a turning point between 3 and 4 agents, and the difference between the behaviours grows the further away the amount of agents is from this point. It can also be seen that the amount of deliveries made decreases as the amount of agents increases. This effect is very slight with circling behaviour, but with language behaviour it can be clearly seen in the figure.

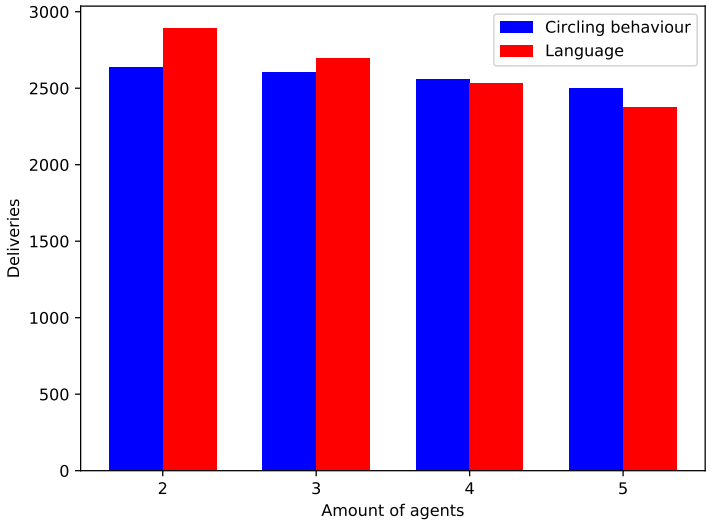


Figure 17: The average amount of deliveries an agent makes in different agent societies.

The reason for the larger decline in deliveries made with language behaviour can be seen in Figure 18. As the amount of agents increases, more collisions happen in the "corridors".

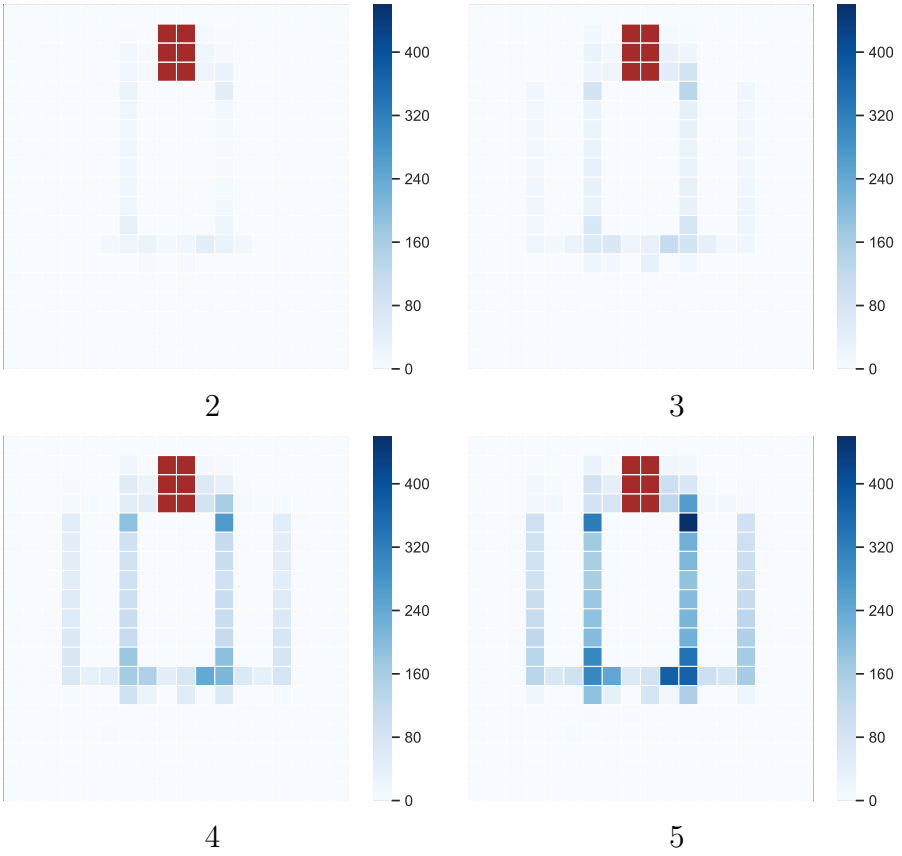


Figure 18: Collision heat maps for societies with different amounts of agents and language behaviour.

7.2.2 Discussion

In the previous experiments, it was seen that at least in some cases language should affect conceptualisation. In these experiments a similar relationship is seen between the environment and behaviour. It was seen that between two behaviour models, one utilising language and the other the programmer's knowledge and understanding of the environment, neither is strictly better in all cases. Of course if the language and its usage were implemented differently, the circling behaviour might emerge as a consequence of the language as well.

Which behaviour model performs better was observed to depend on at least one environment variable, i.e. the amount of agents. When there are 3 or less agents, the language utilising agents are able to often pick the shorter route to the shelves and back. With more agents the "corridors" are occupied more frequently, and the language utilising agents are not able to utilise the shorter route frequently enough to outperform the circling behaviour. With circling behaviour the agents always use the longer route when going to a shelf, and the shorter route when going to the action centre. Although with circling behaviour the agents use the longer route 50% of the time, the decrease in performance due to this is offset by the fact that collisions are eliminated. Other environment variables, such as the layout, are also likely to have different effects on different behaviour models.

The above highlights the importance of not only learning what to communicate, but also how to behave and when to communicate. If the agents are able to learn different behaviours, there might be situations where communication through language is unnecessary. Language learning could for example be combined with reinforcement learning, which is a well established field for learning behaviour. In reinforcement learning agents learn through experience to take actions in order to maximise the reward they gain [24].

The importance of learning when to communicate was also detected in some early experiments that are not reported in this thesis. In the experiments, the agents managed to learn a coherent language, and used it successfully to avoid collisions. Regardless, the amount of deliveries made actually decreased. The reason was that in avoiding collisions, the agents had to take longer paths, which was worse on average than just risking the collision. In this case it would have been better to choose not to communicate at all.

Improvements to the language might make language utilising agents perform better or as well as agents with circling behaviour with all agent amounts. A major limitation in the current implementation is, that the agents do not have any way to understand or communicate their movement direction in the "corridors". If two agents are travelling in the same direction in the same "corridor", there is no danger of collision between them.

The more tools the agents have for understanding the world, the more potential language also has. Increased complexity in conceptualising the world might increase the required complexity of the language, making language emergence more difficult. However increased "cognitive" abilities might also make language emergence easier,

if the agents are able to make better inferences on what another agent might be trying to say. For example, Steels and Loetzsch [23] showed that understanding other agent's visual perspective can be beneficial for language emergence.

7.3 Meaning transfer experiment

The reason for learning general meanings for places and coordinates instead of learning names for specific places in the environment, was that they can be utilised when the environment changes. To test this, the agents first learn a language in a training environment, after which they are moved to a testing environment. See Figure 19 for the environments.

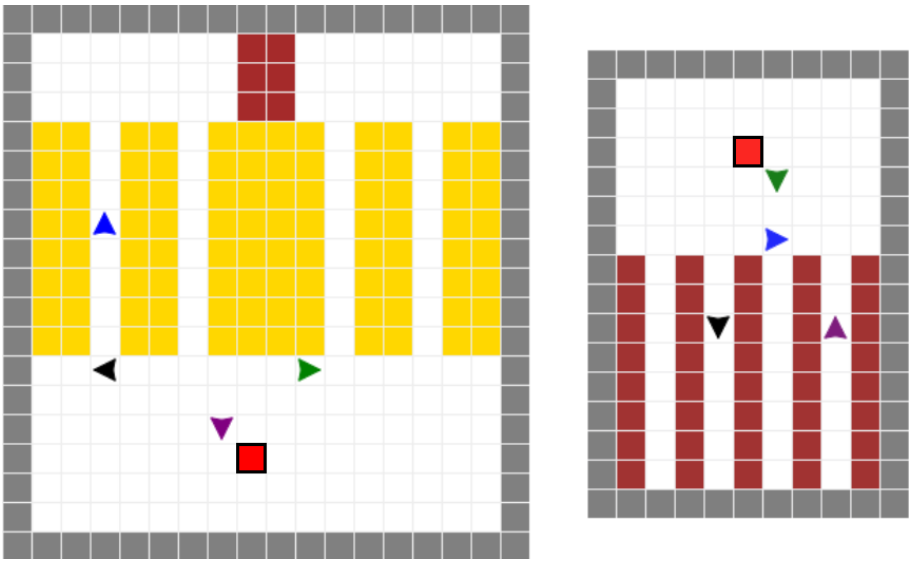


Figure 19: The training environment on the left and the testing environment on the right.

The importance of normalising the coordinates within the context now becomes clear. The absolute distance between the "corridors" in the training and testing environments is different. In the training environment, the absolute range between the leftmost and rightmost "corridors" is also larger. Normalisation allows the agents to use the same meanings in both environments.

Simulations were run with 0, 5000, 20 000, and 50 000 time steps in the training environment, before the society of 4 agents was moved to the testing environment.

The testing environment was run for 20 000 time steps in all cases. 100 simulations were run for each training step amount, and the averages from the testing environment are reported. Therefore, for example the deliveries made in the training environment do not count towards the reported amount of deliveries.

7.3.1 Results

The amount of deliveries with 0 training steps is 1037 ± 2 and with 50 000 training steps 1042 ± 2 (shown with 99% confidence intervals). Surprisingly training before the testing environment doesn't result in a signifigant increase in deliveries made.

Figure 20 shows how fast a language emerges in the testing environment with different amounts of training steps in the training environment. A noticeable difference can be observed in the lines for 0, 5000, and 10 000 training steps until they converge around 13 000 time steps. With 10 0000, 20 000, and 50 000 training steps a difference can be seen only in the very beginning, after which the lines converge around 2000 time steps. Overall it can be seen that more training steps in the training environment results in faster language emergence.

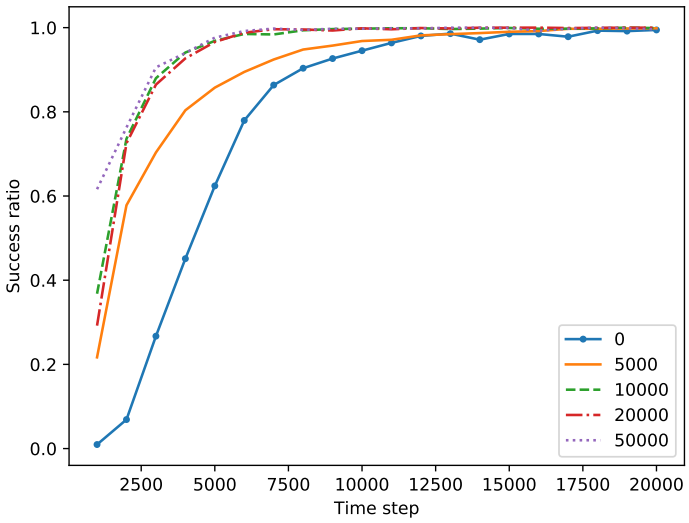


Figure 20: Complete success ratios in the testing environment for agent societies with different amounts of training steps. The lines are smoothed over 1000 time steps.

Only discrimination meanings transfer from the training environment, as the "corri-

dors" are different in the environments. To analyse how well they transfer, part 1 of the Query game (see Section 6.3) is inspected. In it the speaker utters a word for a place meaning, and another word for a discrimination meaning. Table 4 shows that with 0 training steps, most failures are caused by misinterpretation of the discrimination word. As the amount of training steps increases, less games fail because of the discrimination word, and a higher and higher portion of failures are caused by misinterpretation of the place word. With 50 000 steps the agents have a coherent language for the discrimination meanings, which transfers perfectly to the testing environment. The language in the testing environment emerges fast, because the agents only need to learn a place meaning.

	Misinterpretation % for		
Training steps	Discrimination word	Place word	Failed games
0	95.0%	12.4%	668
5000	78.6%	27.5%	199
10 000	32.9%	68.7%	52
20 000	0.5%	99.6%	48
50 000	0.0%	100.0%	45

Table 4: Percentage of failed games where at least one agent misinterpreted the discrimination or place word. Also showing the total amount of failed games.

7.3.2 Discussion

These experiments show that at least the discrimination meanings can be transferred to new environments. The author sees no reason why place meanings wouldn't transfer similarly. However, some problems might emerge regarding the importance values that are associated with the place meanings. A place meaning that has been learned to be important in an old environment might not be important in the new environment and vice versa.

Learning the discrimination meanings in the training environment didn't improve performance in any significant manner in the testing environment, because the language emerges so quickly from scratch. In a situation where the language emergence is slower the outcome would probably be different. Possible causes for a slower language emergence include a more complex language, larger society, and meanings

that are harder to learn.

7.4 Summary

Three experiments were conducted to explore different aspects of the language. In the route conceptualisation experiment it was shown how the agents could utilise their learned importance values to conceptualise their route options. Also with score filtering the importance of feedback from the language into how the agents conceptualise the world was shown.

The circling behaviour experiment showcased the importance of taking a holistic view to language learning. When and what the agents should communicate about depends on the environment and the behaviour of the agents.

The meaning transfer experiment showed that the agents can successfully transfer discrimination meanings into new environments. However, no significant improvement was detected in performance in this particular experiment setting, although improvements were seen in language emergence speed.

8 Related work

Language games are a popular tool for researching emergent communication amongst computational agents [25]. They are commonly used to analyse different emergence mechanisms, meaning and lexicon structures, and aspects of language. For example, Nevens and Spranger [26] investigate tutor feedback in language learning. In their experiments a tutor agent is teaching a language to a learning agent. The tutor can give feedback on a language game by pointing, which speeds up the learning. Lazaridou et al. [27] use neural networks as implicit meaning structures. Their agents play a language game where the agents see two images, one of which is the topic. The speaker uses a neural network to map the topic (given the context) to a symbol in the agents' vocabulary. The hearer uses a neural network to map the symbol and the context to one of the two images, guessing which is the topic. Cuskley et al. [28] study dynamics of regularity and irregularity in language systems. Natural language has many regular rules, which have irregular exceptions. For instance, the past tense for preach is preached (regular) and for teach is taught (irregular).

In contrast the focus of the work presented here is on emergence of language that can be used as a tool for some purpose. The properties of the language, its emergence,

and even communication success are secondary to whether the agents can improve their performance (see Section 3).

Emergent communication research is not limited to language games. E.g. in the experiments of Mordatch and Abbeel [3], the agents utter communication symbols at each time step. The agents don't participate in any formal language games and the uttered symbols are only part of what an agent can observe about its environment. Although goal monitoring and option-based discrimination were used alongside language games in this thesis, they are not dependent on language games. It could be argued, that Mordacth and Abbeel use a form of goal monitoring, as their agents learn a language based on a shared reward, which reflects how well the whole society is performing.

In recent years neural networks have been popular in emergent communication research [2, 3, 4, 29, 27, 30]. Their black box nature makes it hard to analyse what kind of meanings the agents learn, and how they are associated with verbal utterances. Also their training is slow and requires a lot of data. Neural networks were not used here, because for the purposes of this thesis, simpler, explicit models seemed more appropriate. The focus of the research presented here is not on the language models themselves, but the introduced methods that can be used to ground the language models in practicality. However, these methods are not tied to the language models used in this thesis, and could be used with other models like neural networks as well.

For more information on communication emergence amongst computational agents, the reader is guided to the surveys by Kirby [31] and Wagner [32].

9 Final words

9.1 Conclusions

The goal of this thesis was to investigate how language could be grounded in practicality in a society of agents playing language games. The thesis started with a look into the problem of grounding language in perception, as it would serve as a basis for practicality grounding.

The contributions of this thesis began by extending the problem of grounding language in perception to include grounding in practicality. The main idea behind practicality grounding is that the needs of the agents should guide what kind of

a language emerges and how it is used. To enable grounding language in practicality three novel methods were introduced: goal monitoring, using importance to determine a game's context, and option-based discrimination.

To facilitate emergence of the language two new language games were introduced: the Place Game and the Query Game. These games were designed for learning place and discrimination meanings and words for them. The methods for grounding language in practicality were used to guide what the agents communicate in the Query Game, so that they could share useful information with each other.

Empirical experiments were conducted in a simulated warehouse environment, where the agents' task was to fetch items from shelves. The experiments showed that a language that improves the agents' performance, i.e. a practical language, can emerge as a consequence of the language games and methods introduced in this thesis. Additional experiments were run to investigate various aspects of the language.

The work presented here focuses on very elementary aspects of grounding language in practicality. The author hopes that the rudimentary methods and experiments presented here could serve as a starting point and inspiration for others to continue the research.

9.2 Future work

Learning language in a setting where the agents act to perform tasks or reach goals means that the language is tied to everything the agents do or reason about. This raises several questions, such as when should an agent communicate [33]. Some of the possible directions of future research are discussed here.

This thesis was based on early language game research. Since then the field has moved on and several developments have happened that could be revisited from practicality's perspective. These include using grammar [34] and semiotic networks [35].

In the experiments the agents' behaviour and language was based on a 2D map. Although the agents had the same map of the environment, the presented methods should work in theory even if each agent built its own map. Because of how the context is derived from the map and normalised, the agents don't need a common origin for their personal maps. Therefore the research conducted here might be relatively easily continued on real robots utilising simultaneous localization and mapping (SLAM) [36]. In SLAM agents simultaneously build a map of an unknown environment and track their location in it.

Computer simulations have been used to research language emergence not only in computational agents, but humans as well [37]. In the author's mind it would be fairly safe to state that humans use language for practical purposes, and so simulations investigating practical language emergence could be helpful in understanding the evolution of human language.

Another interesting direction to take would be investigating the possibilities of computational creativity for language emergence. Colton and Wiggins [38] define computational creativity research as: "The philosophy, science and engineering of computational systems which, by taking on particular responsibilities, exhibit behaviours that unbiased observers would deem to be creative." In the field of computational creativity language games have been used to evolution of language in creative domains [39, 15, 40]. It would be interesting to apply computational creativity techniques to the creation of language itself. For example, instead of randomly generating new words as was done in this thesis, the words could be created in some meaningful way so that their form already communicates something.

The agents in this thesis don't model each other's "minds" in any way. Not only could this kind of modelling be beneficial for cooperation between the agents in general [8], but it could help language emergence. For example, understanding other agents' visual perspective of the environment can aid language emergence [23]. A natural continuation for the work presented in this thesis would be modelling other agents' goals.

Combining reinforcement learning with language learning was already discussed in Section 7.2. This is a promising direction for future research, as the goal of reinforcement learning and practical language is the same, i.e. learning to act better in the world. The method for updating the importance values, which dictated what the agents communicated about, was already inspired by a reinforcement learning method called Q-learning [22].

References

1 W. V. Quine, *Word and Object.*, vol. New ed. The MIT Press, 2013.

2 J. Foerster, I. A. Assael, N. de Freitas, and S. Whiteson, "Learning to communicate with deep multi-agent reinforcement learning," in *Advances in Neural*

Information Processing Systems 29 (D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, eds.), pp. 2137–2145, Curran Associates, Inc., 2016.

3 I. Mordatch and P. Abbeel, "Emergence of grounded compositional language in multi-agent populations," in *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

4 S. Sukhbaatar, R. Fergus, *et al.*, "Learning multiagent communication with backpropagation," in *Advances in Neural Information Processing Systems*, pp. 2244–2252, 2016.

5 L. Steels, *The Talking Heads experiment: Origins of words and meanings*, vol. 1. Berlin: Language Science Press, 2015.

6 O. Hantula, "On grounding language games in practicality," in *AISB Language Learning for Artificial Agents symposium*, pp. 16–20, 2019.

7 Y. Shoham and K. Leyton-Brown, *Multiagent systems: Algorithmic, game-theoretic, and logical foundations*. Cambridge University Press, 2008.

8 C. Castelfranchi, "Modelling social action for ai agents," *Artificial intelligence*, vol. 103, no. 1-2, pp. 157–182, 1998.

9 J. Bleys, M. Loetzsch, M. Spranger, and L. Steels, "The grounded colour naming game," in *Proceedings Spoken Dialogue and Human-Robot Interaction workshop at the RoMan 2009 conference*, Institute of Electrical and Electronics Engineers, 2009.

10 *Oxford Dictionary of English (3 ed.)*. Oxford University Press, 2010.

11 L. Steels and F. Kaplan, "Bootstrapping grounded word semantics," in *Linguistic Evolution through Language Acquisition: Formal and Computational Models* (T. Briscoe, ed.), pp. 53–73, Cambridge: Cambridge University Press, 2002.

12 L. Wittgenstein, *Philosophical Investigations*. Oxford: Blackwell, 1953.

13 L. Steels, "A self-organizing spatial vocabulary," *Artificial life*, vol. 2, no. 3, pp. 319–332, 1995.

14 L. Steels, "The spontaneous self-organization of an adaptive language.," in *Machine Intelligence 15*, Oxford University Press, 1996.

15 R. Saunders, "Artificial creative systems and the evolution of language.," in *ICCC*, pp. 36–41, 2011.

16 A. D. Smith, "Establishing communication systems without explicit meaning transmission," in *European Conference on Artificial Life*, pp. 381–390, Springer, 2001.

17 T. Lindh-Knuutila, T. Honkela, and K. Lagus, "Simulating meaning negotiation using observational language games," in *Symbol grounding and beyond*, pp. 168–179, Springer, 2006.

18 M. Spranger, M. Loetzsch, and L. Steels, "A perceptual system for language game experiments," in *Language grounding in robots*, pp. 89–110, Springer, 2012.

19 L. Steels, "Perceptually grounded meaning creation," in *Proceedings of the International Conference on Multiagent Systems (ICMAS-96)*, pp. 338–44, 1996.

20 P. E. Hart, N. J. Nilsson, and B. Raphael, "A formal basis for the heuristic determination of minimum cost paths," *IEEE transactions on Systems Science and Cybernetics*, vol. 4, no. 2, pp. 100–107, 1968.

21 C. Claus and C. Boutilier, "The dynamics of reinforcement learning in cooperative multiagent systems," *AAAI/IAAI*, vol. 1998, pp. 746–752, 1998.

22 C. J. Watkins and P. Dayan, "Q-learning," *Machine learning*, vol. 8, no. 3-4, pp. 279–292, 1992.

23 L. Steels and M. Loetzsch, "Perspective alignment in spatial language," in *Spatial Language and Dialogue* (K. R. Coventry, T. Tenbrink, and J. Bateman, eds.), Oxford University Press, 2009.

24 R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 2018.

25 J. Nevens, P. Van Eecke, and K. Beuls, "A practical guide to studying emergent communication through grounded language games," in *AISB Language Learning for Artificial Agents symposium*, pp. 1–8, 2019.

26 J. Nevens and M. Spranger, "Computational models of tutor feedback in language acquisition," in *2017 Joint IEEE International Conference on Development and Learning and Epigenetic Robotics (ICDL-EpiRob)*, pp. 221–226, IEEE, 2017.

27 A. Lazaridou, A. Peysakhovich, and M. Baroni, "Multi-agent cooperation and the emergence of (natural) language," *arXiv preprint arXiv:1612.07182*, 2016.

28 C. Cuskley, C. Castellano, F. Colaiori, V. Loreto, M. Pugliese, and F. Tria, "The regularity game: Investigating linguistic rule dynamics in a population of interacting agents," *Cognition*, vol. 159, pp. 25–32, 2017.

29 S. Havrylov and I. Titov, "Emergence of language with multi-agent games: learning to communicate with sequences of symbols," in *Advances in Neural Information Processing Systems*, pp. 2149–2159, 2017.

30 A. Lazaridou, K. M. Hermann, K. Tuyls, and S. Clark, "Emergence of linguistic communication from referential games with symbolic and pixel input," in *International Conference on Learning Representations*, 2018.

31 S. Kirby, "Natural language from artificial life," *Artificial life*, vol. 8, no. 2, pp. 185–215, 2002.

32 K. Wagner, J. A. Reggia, J. Uriagereka, and G. S. Wilkinson, "Progress in the simulation of emergent communication and language," *Adaptive Behavior*, vol. 11, no. 1, pp. 37–69, 2003.

33 A. Singh, T. Jain, and S. Sukhbaatar, "Learning when to communicate at scale in multiagent cooperative and competitive tasks," in *International Conference on Learning Representations*, 2019.

34 R. van Trijp, *The evolution of case grammar*. Berlin: Language Science Press, 2016.

35 L. Steels, M. Spranger, R. van Trijp, S. Höfer, and M. Hild, "Emergent action language on real robots," in *Language Grounding in Robots* (L. Steels and M. Hild, eds.), pp. 255–276, Springer, 2012.

36 C. Cadena, L. Carlone, H. Carrillo, Y. Latif, D. Scaramuzza, J. Neira, I. Reid, and J. J. Leonard, "Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age," *IEEE Transactions on robotics*, vol. 32, no. 6, pp. 1309–1332, 2016.

37 M. H. Christiansen and S. Kirby, "Language evolution: Consensus and controversies," *Trends in cognitive sciences*, vol. 7, no. 7, pp. 300–307, 2003.

38 S. Colton, G. A. Wiggins, *et al.*, "Computational creativity: The final frontier?," in *Ecai*, vol. 2012, pp. 21–16, Montpelier, 2012.

39 R. Saunders and K. Grace, "Towards a computational model of creative cultures.," in *AAAI Spring Symposium: Creative Intelligent Systems*, pp. 67–74, 2008.

40 A. Zhang and R. Saunders, "Exploring conceptual space in language games using hedonic functions," in *ICCC*, pp. 276–279, 2014.